Generalized Model Predictive Pulse Pattern Control Based on Small-Signal Modelling

Martinus David Dorfling

Dissertation presented in fulfilment of the requirements for the degree of Doctor of Philosophy in (Electrical) Engineering in the Faculty of Engineering at Stellenbosch University

Supervisor: Prof Hendrik du Toit Mouton
Co-supervisor: Prof Tobias Geyer

March 2021

Declaration

By submitting this thesis electronically, I declare that the entirety of the work contained therein is my own, original work, that I am the sole author thereof (save to the extent explicitly otherwise stated), that reproduction and publication thereof by Stellenbosch University will not infringe any third party rights and that I have not previously in its entirety or in part submitted it for obtaining any qualification.

Tinus Dorfling

March 2021

Abstract

Optimized pulse patterns (OPPs) are a pulse-width modulation method in which the switching pattern is computed offline. Typically, the harmonic distortions for a given switching frequency are minimized. OPPs are particularly beneficial for industrial power electronic systems that operate at low switching frequencies (such as medium-voltage drive systems). However, designing a controller with a high dynamic performance for higher-order converter systems that are modulated by OPPs is a difficult and somewhat unexplored task. For first-order converter systems, a state-of-the-art industrial control technique known as model predictive pulse pattern control achieves a high dynamic performance.

This thesis proposes a generalized model predictive pulse pattern controller that is applicable to (linear) higher-order converter systems. Using the notion of small-signal modelling, the dynamic equations of the state variables of the converter system are linearized around the optimal steady-state trajectory that results from the OPP. Key to the control method is to model the modifications to a pulse pattern with the strengths of impulses, resulting in the modifications to the converter states being linear in the impulse strengths. The proposed controller is formulated according to the model predictive control methodology. Thanks to the linear internal dynamic model, the underlying optimization problem can be formulated as a convex quadratic program. Simulation results demonstrate that the proposed controller achieves a very short response time during transients and superior harmonic performance during steady-state operation. Importantly, an implementation of the control algorithm on a low-cost field-programmable gate array demonstrates that the controller can execute in real-time within a short sampling interval of 25 μs; thus far, none of the (few) existing OPP-based controllers for higher-order converter systems have been proven to be practically implementable.

Additionally, the control method is augmented with constraints on the state variables. Specifically, the state variables are given bounds that they should remain within. The method is verified through simulation. Furthermore, balancing of the neutral-point potential is integrated in the controller. Simulation results show that the balancing method performs well under dynamic operating conditions, including during zero power factor at the converter terminals, where traditional balancing methods tend to fail.

Opsomming

Geoptimeerdepulspatrone (OPPs) is ’n pulswydte-modulasiemetode waarin die skakelpatroon vanlyn bereken word. Die harmoniese distorsie van ’n gegewe skakelfrekwensie sal tipies geminimeer word. OPPs is veral voordelig vir industriële drywingselektroniese stelsels wat teen ’n lae skakelfrekwensie funksioneer (soos, byvoorbeeld, ’n mediumspanning-aandrywingstel). Om ’n beheerder met ’n hoë dinamiese optrede te ontwerp vir hoër-orde omsetterstelsels wat gemoduleer word deur OPPs is egter ’n moeilike taak en nog redelik onverken. Vir eerste-orde omsetterstelsels kan ’n nuwe industriële tegniek, wat bekend staan as modelvoorspelling-pulspatroonbeheer, ’n hoë dinamiese optrede bereik.

Hierdie tesis bied ’n veralgemeende modelvoorspelling-pulspatroonbeheerder wat van toepassing is op (lineêre) hoër-orde stelsels. Deur gebruik te maak van kleinseinmodellering kan die dinamiese vergelykings van die toestandsveranderlikes van die omsetter-stelsel gelineariseer word rondom die optimale bestendigdetoestandtrajek. Dit kom as gevolg van ’n nominale bestendigdetoestandpulspatroon. ’n Belangrike aspek van die beheermetode is om die wysigings aan die pulspatroon te modelleer met die sterkte van impulse. Die gevolg is dat die wysigings aan die omsettertoestande lineêr is in die sterkte van die impulse. Die voorgestelde beheerder word geformuleer volgens die modelvoorspellingsbeheermetodologie. Danksy die lineêre interne dinamiese model kan die onderliggende optimeringsprobleem geformuleer word as ’n konveks-kwadratiese program. Volgens simulasie resultate bereik die voorgestelde beheerder ’n baie kort reaksietyd tydens oorgange en uitmuntende harmoniese optrede tydens bestendigdetoestandwerking. Die implementering van hierdie beheerstelsel op ’n lae-koste veld-programmeerbare hekskikking (FPGA) demonstreer dat die beheerder intyds kan funksioneer binne ’n kort monsterperiode van 25 µs. Tot dusver kon geen van die (min) bestaande OPP-gebasseerde beheerders vir hoër-orde stelsels daarin slaag om prakties uitvoerbaar te wees nie.

Die beheermetode word verder aangepas met beperkings op die toestandsveranderlikes. Die toestandsveranderlikes word spesifiek perke gegee waarbinne hul behoort te bly. Hierdie metode word bevestig deur middel van simulasies. Die balansering van die neutrale-puntspanning word ook geïntegreer in die beheerder. Simulasies toon dat die balanseermetode goed presteer onder dinamiese toestande. Dit sluit in tydens nuldrywingsfaktor by die omsetterterminale waar die tradisionele balanseermetodes neig om te faal.

Acknowledgements

During my undergraduate and postgraduate studies, I have had the wonderful opportunity to work under Prof Toit Mouton. The knowledge you have shared, the skills you have taught, and the opportunities you have presented are greatly appreciated, and invaluable to me. If not for your excellent (and highly enjoyable) power electronics course during undergraduate, I may have never developed an interest for power electronics.

I would like to thank Tobias Geyer for his support and enthusiasm throughout my studies. No matter how busy you were, you always managed to thoroughly address any questions I had, gave highly detailed feedback, and supported me in any way you could. Between you and Prof Mouton, I had access to an encyclopedia of knowledge.

If it not for Stefan Richter's help on optimization, the content of my thesis would surely have been reduced. Thank you for being more than helpful whenever I, a stranger that emailed you, had questions on optimization. Your recommendations saved me more time than you could realise.

I would like to thank ABB for funding this research. A special thanks is owed to Gerald Scheuer. Thank you, and ABB, for allowing my great freedom during my research.

At the age of 27, I am a basement dweller living with my mom, Suna, rent-free and getting home-made food. I know this will soon come to an end, and I will be kicked out of the nest. Baie dankie vir Ma se liefde en ondersteuning deur al die jare.

A very warm, special thanks to Aanch for her unconditional love and support. Thank you for also being my best friend, and being my favourite person. I wish I could add more emotion to my words, but I have no idea who might read it.

Contents

List of Figures

List of Tables

Nomenclature

Abbreviations

ac	Alternating current
CB	Carried-based
dc	Direct current
ESR	Equivalent series resistance
FCS	Finite-control-set
FPGA	Field-programmable gate array
HIL	Hardware-in-the-loop
LQR	Linear-quadratic regulator
MIMO	Multiple-input multiple-output
MPC	Model predictive control
MP^3C	Model predictive pulse pattern control
NPC	Neutral-point-clamped
OPP	Optimized pulse pattern
PCC	Point of common coupling
pu	Per unit
PWM	Pulse-width modulation
QP	Quadratic program
rms	Root-mean-square
TDD	Total demand distortion

Variables

$z \in \mathbb{R}$	scalar
$\boldsymbol{z} \in \mathbb{R}^n$	column vector with dimension n
$\boldsymbol{M} \in \mathbb{R}^{n \times m}$	matrix with dimensions $n \times m$
$\mathcal{S}$	set

Symbols

$\mathbf{0}_n$	n-dimensional (column) vector of zeros
$\mathbf{0}_{n \times n}$	$n \times m$ matrix of zeros
$\mathbf{1}_n$	n-dimensional (column) vector of ones
$\mathbf{1}_{n \times n}$	$n \times m$ matrix of ones
$\mathbf{I}_n$	identity matrix of dimensions n
$\boldsymbol{A}$	constraint matrix
$\boldsymbol{b}$	constraint vector
$\boldsymbol{c}$	vector with linear coefficients
C	filter capacitance
C_d	half dc-link capacitance
d	pulse number
$\boldsymbol{F}$	state matrix
$\boldsymbol{G}$	input matrix
γ	relaxation weight
$\boldsymbol{H}$	Hessian matrix (simply referred to as the Hessian)
$\boldsymbol{K}$	Clarke transformation matrix
$\boldsymbol{K}^{-1}$	inverse Clarke transformation matrix
$\lambda_{p,i}$	strength of the ith impulse of phase p
$\boldsymbol{\Lambda}$	strength vector (the vector of impulse strengths $\lambda_{p,i}$)
L_c	Lipschitz constant
L	filter inductance
L_g	grid inductance
L_t	transformer leakage inductance
m_a	modulation index
$\boldsymbol{P}$	disturbance matrix
$\boldsymbol{Q}$	penality matrix for state variables
q_{np}	penality on neutral-point potential
R	filter resistance
R_g	grid resistance
R_t	transformer resistance
$\boldsymbol{R}$	penality matrix for switching time modifications
s	step size
σ	limit on a converter quantity
$\Delta\sigma$	relaxation variable
t	time
$t_{p,i}$	ith modified switching instant of phase p
$t'_{p,i}$	ith incumbent switching instant of phase p
$t^*_{p,i}$	ith nominal switching instant of phase p
$\Delta t_{p,i}$	ith switching instant modification of phase p, $t_{p,i} - t'_{p,i}$

T_p	prediction horizon
T_c	constraint prediction interval
T_s	sampling interval
u	switch position
Δu_i	ith switching transition, $u_i - u_{i-1}$
$\boldsymbol{u}_{abc}$	three-phase switch position, or incumbent pulse pattern
$\boldsymbol{u}^*_{abc}$	nominal three-phase pulse pattern
$\boldsymbol{u}_{abc,\text{mod}}$	modified three-phase pulse pattern
$\boldsymbol{u}'_{abc}$	three-phase absolute incumbent pulse pattern
$\tilde{\boldsymbol{u}}_{abc}$	small-signal input
$\boldsymbol{v}_{abc}$	three-phase converter voltage
$\boldsymbol{v}_g$	grid voltage
v_n	neutral-point potential
$\tilde{v}_n$	small-signal neutral-point error
V_d	dc-link voltage
$V_{g,\text{LL}}$	line-to-line rms grid voltage
$\boldsymbol{x}$	state vector
$\boldsymbol{x}^*$	steady-state trajectory
$\tilde{\boldsymbol{x}}$	small-signal error

Operations

$z \in \mathcal{S}$	z belongs to $\mathcal{S}$
$\boldsymbol{z}^{\mathrm{T}}$	transpose of the (column) vector $\boldsymbol{z}$
$\lvert\boldsymbol{z}\rvert$	$\lvert\boldsymbol{z}\rvert = [\lvert z_1\rvert \ \lvert z_2\rvert \ \cdots \ \lvert z_n\rvert]^{\mathrm{T}}$, componentwise absolute value of the vector $\boldsymbol{z}$
$\lVert\boldsymbol{z}\rVert_2$	$\lVert\boldsymbol{z}\rVert_2 = \sqrt{\boldsymbol{z}^{\mathrm{T}}\boldsymbol{z}} = \sqrt{\sum_{i=1}^{n} z_i^2}$, 2-norm (or Euclidean norm) of vector $\boldsymbol{z}$
$\lVert\boldsymbol{z}\rVert^2_{\boldsymbol{M}}$	$\lVert\boldsymbol{z}\rVert^2_{\boldsymbol{M}} = \boldsymbol{z}^{\mathrm{T}}\boldsymbol{M}\boldsymbol{z}$, 2-norm squared of vector $\boldsymbol{z}$ weighted with matrix $\boldsymbol{M}$
$\nabla f(\boldsymbol{z})$	$\nabla f(\boldsymbol{z}) = [\frac{\partial f(\boldsymbol{z})}{\partial z_1} \ \frac{\partial f(\boldsymbol{z})}{\partial z_2} \ \cdots \ \frac{\partial f(\boldsymbol{z})}{\partial z_{n_z}}]^{\mathrm{T}}$, the gradient (vector) of the function f
$\pi_{\mathcal{Z}}(\boldsymbol{x})$	projection operator that projects the vector $\boldsymbol{x}$ on the set $\mathcal{Z}$
$\boldsymbol{M}^{\mathrm{T}}$	transpose of the matrix $\boldsymbol{M}$
$\boldsymbol{M}^{-1}$	inverse of the matrix $\boldsymbol{M}$
$\lVert\boldsymbol{M}\rVert_2$	$\lVert\boldsymbol{M}\rVert_2 = \max_{\boldsymbol{z}\neq 0} \frac{\lVert\boldsymbol{M}\boldsymbol{z}\rVert_2}{\lVert\boldsymbol{z}\rVert_2}$, (induced) 2-norm of matrix $\boldsymbol{M}$
$\lVert\boldsymbol{M}\rVert_1$	$\lVert\boldsymbol{M}\rVert_1 = \max_{1\leq j\leq n} \sum_{i=1}^{n} \lvert\boldsymbol{M}_{(i,j)}\rvert$, 1-norm of matrix $\boldsymbol{M}$
$\lVert\boldsymbol{M}\rVert_\infty$	$\lVert\boldsymbol{M}\rVert_\infty = \max_{1\leq i\leq n} \sum_{j=1}^{n} \lvert\boldsymbol{M}_{(i,j)}\rvert$, infinity-norm of matrix $\boldsymbol{M}$

Part I

Introduction

Chapter 1

Introduction

This chapter serves as an introduction to this thesis. First, a background on the medium-voltage power electronics industry is given. This includes a review of the requirements of control methods for medium-voltage applications. The control problem that needs addressing is then identified. This is followed by a summary of the contributions of this thesis. The chapter concludes with the outline of this thesis and a brief summary of each chapter.

Chapter Contents

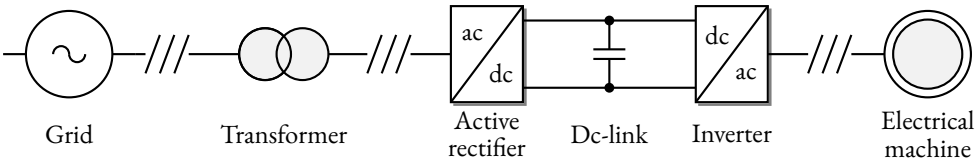


Figure 1.1: Variable-speed drive system.

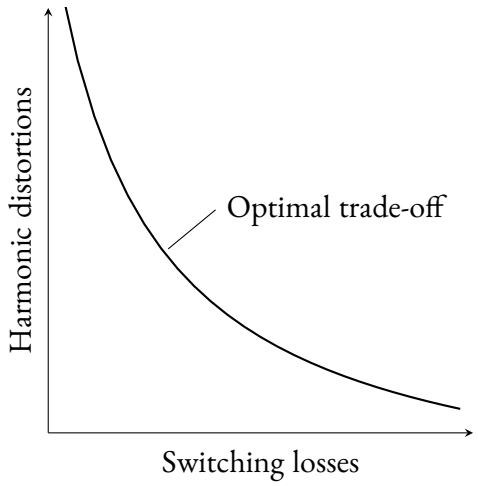


Figure 1.2: The fundamental trade-off between harmonic distortions and switching losses in power electronics.

1.1 Background

The medium-voltage power electronics industry involves converter systems with power ratings in excess of 1 MVA. Arguably, the most common medium-voltage application is a variable-speed drive. A block diagram of such a system is shown in Figure 1.1. Typically, a back-to-back converter configuration is used: a grid-connected converter establishes a dc-link voltage, which a converter connected to an electrical machine uses as its voltage source. In this thesis, only the grid-connected side is considered. For a medium-voltage converter system, there are three main requirements regarding its control and modulation method: low harmonic distortions, low switching losses, and high controller bandwidth.

The first two requirements, low harmonic distortions and low switching losses (that is, low harmonic distortions per switching losses), are interconnected. In power electronics, there is a fundamental trade-off between the harmonic distortions and switching losses (which are proportional to the device switching frequency) of a converter system, see Figure 1.2. Typically, if only one of these objectives is prioritized, the other is compromised. Instead of optimizing for one of the two objectives, the *optimal trade-off* point should be moved closer to the origin since both objectives are beneficial. Lower harmonic distortions result in smaller (and therefore cheaper) filter components being required. Moreover, grid-connected converters are subject to harmonic standards that have to be satisfied. On the other hand, decreasing the power losses allows for the power rating of a converter system to increase (and thus its selling price as well). Due to the high-power application of medium-voltage systems, with voltages and currents in the kilovolt and kiloampere range, high-power semiconductors are required, for which the switching losses are significant. Often, they are of the same magnitude as the conduction losses. Therefore, the switching frequency of the semiconductor devices is typically limited to a few hundred hertz.

Furthermore, a converter system is often required to react fast to changes in operating conditions, such as reference steps or faults. This is achieved by having a controller with a high bandwidth

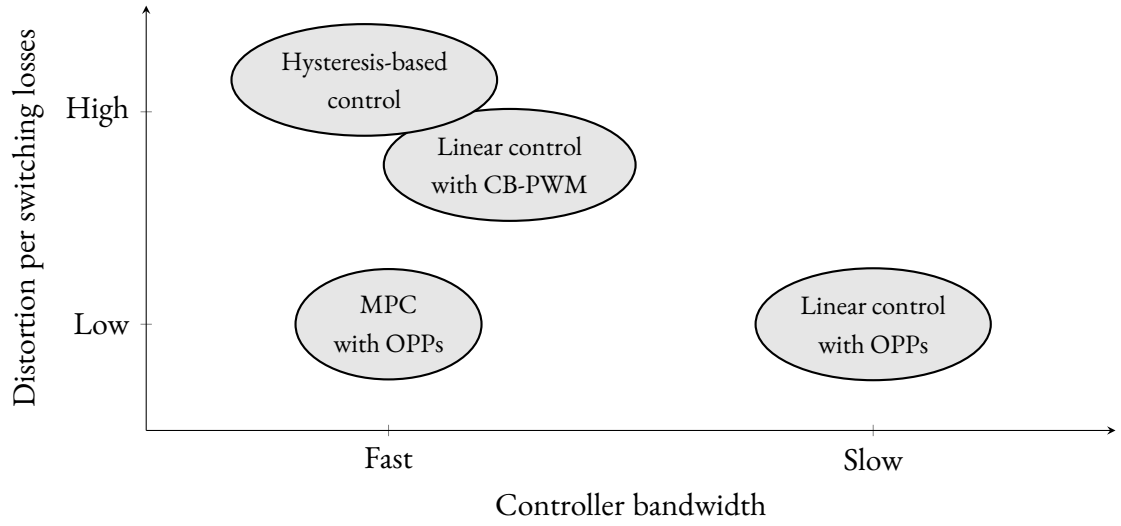


Figure 1.3: Classification of control methods for medium-voltage applications. Adapted from [4, Figure 1.4].

(which results in a short response time). However, having a high-bandwidth controller in addition to low harmonic distortions per switching losses does pose a significant challenge; very few control methods satisfy this criteria. Hysteresis-based controllers, such as direct power control [1], use lookup tables to select the appropriate switching state of a converter. These methods have a rapid response. However, the harmonic distortions are typically high. Another group of control methods that have a short response time are linear controllers (such as PI controllers) in conjunction with carrier-based pulse-width modulation (CB-PWM). An example of such a technique is field voltage-oriented control [2]. Unfortunately, methods that use CB-PWM as a modulation technique at low switching frequencies typically suffer from high harmonic distortions. In order to achieve a near optimal ratio of harmonic distortions per switching frequency, a modulation technique known as optimized pulse patterns (OPPs) can be used. OPPs are offline-calculated switching patterns that minimize the harmonic distortions of a given system. However, designing a high-bandwidth controller for OPPs is a difficult task; typically, a linear controller with a low bandwidth is used.

Recently in 2012, the control problem underlying OPPs was formulated in a model predictive control framework, giving rise to model predictive pulse pattern control (MP^{3}C) [3]. The controller combines a short response time (similar to that of hysteresis-based controllers) during transients with the superior steady-state harmonic performance of OPPs. MP^{3}C is being utilized in the modern industrial drive systems of ABB. In Figure 1.3, the aforementioned control methods are classified according to their harmonic performance and dynamic response.

However, MP^{3}C was originally designed for the control of electrical machines. Specifically, MP^{3}C is only applicable to first-order systems that can be modelled as an integrator (such as an inductive load). A new high-bandwidth control method, that is practically implementable, is required that addresses the control of higher-order converter systems that are modulated by OPPs.

1.2 Thesis Contributions

The primary contribution of this thesis is a new OPP-based model predictive controller that is applicable to higher-order systems. Specifically, the control method addresses the control problem of linear multiple-input multiple-output (MIMO) converter systems that are modulated by OPPs. In a sense, the control method is a generalization of MP^{3}C. The control method regulates the state vector of a converter system along the steady-state trajectory that results from an OPP. Specifically,

the control method modifies the (nominal) pulse pattern during transients (such as reference steps, faults, or disturbances) to achieve fast closed-loop control. Then, during steady-state operation, the control method modulates the converter system with the nominal pulse pattern to achieve the superior harmonic performance of OPPs. A patent application for this control method has been filed by ABB [5].

Furthermore, the formulation of the proposed controller is extended to include constraints on the state vector in the form of bounds. The control method is further extended to include the balancing of the neutral-point potential of the neutral-point-clamped converter. Thus far, the neutral point balancing problem has not been completely solved for pulse patterns, as traditional balancing methods tend to fail when operating under zero power factor at the converter terminals.

Finally, the proposed control method (without constraints on the state vector and balancing of the neutral-point potential) is implemented on a low-cost field-programmable gate array (FPGA) in order to prove its practical feasibility. None of the (few) existing OPP-based controllers for higher-order systems have been shown to execute in real-time operation.

In summary, the contributions of this thesis are

- a model predictive pulse pattern controller for linear MIMO converter systems that are modulated by OPPs,
- the extension of the formulation of the control method so that bounds can be imposed on the state vector,
- the integration of the balancing of the neutral-point potential in the controller, and
- an (efficient) implementation of the control algorithm on a low-cost FPGA.

1.3 Thesis Outline and Summary

This thesis is arranged in three parts.

Part I: Introduction serves as an introduction and reviews the theory required for this thesis. Existing work is also reviewed.

Chapter 2 gives a brief overview on mathematical optimization. This includes the required notation and problem statements. Specific attention is given to quadratic programs, which frequently arise throughout this thesis. Some details are presented on optimization algorithms, which are numerical methods employed to solve optimization problems. The gradient projection method is highlighted and thoroughly discussed. Illustrative examples are given throughout the chapter.

Chapter 3 discusses power electronics and optimized pulse patterns. Initially, a few preliminary concepts regarding power electronics and power systems are given. This includes a review on three-phase systems, the Clarke transformation, and the per-unit system. The neutral-point-clamped converter is then presented. The application of grid-connected converters is briefly discussed, which is followed by a model for a grid-connected converter. The chapter concludes with a thorough introduction to OPPs, which is the modulation technique used throughout this thesis. To demonstrate the effectiveness of OPPs, a comparison with well-known CB-PWM is presented.

Chapter 4 gives an introduction to model predictive control and reviews existing control techniques. First, the origins and underlying control principle of model predictive control are given. Then, finite-control-set model predictive control, a popular control technique used in the power electronics research community, is briefly reviewed. The remainder of the chapter discusses control techniques that are employed to address the control problem underlying OPPs. The concept of

MP^{3}C is discussed in moderate detail, including its formulation. The chapter concludes with a review of OPP-based control techniques for higher-order systems; the shortcomings of these techniques are also discussed.

Part II: Generalized Model Predictive Pulse Pattern Control Based on Small-Signal Modelling presents the research contribution of this thesis.

Chapter 5 presents the so-called small-signal controller, which is a generalized model predictive pulse pattern controller that is applicable to any linear MIMO converter system that is modulated by OPPs. The notion of small-signal modelling is first explained. Then, a method to determine the steady-state trajectory of a converter system that is modulated by a pulse pattern is derived. The linearization to efficiently model modifications of a pulse pattern is explained. This is followed by the derivation of the small-signal controller, which combines OPPs with a model predictive controller. A performance evaluation, through use of simulation, demonstrates that the controller exhibits high dynamic performance during transients, and applies the nominal pulse pattern during steady-state conditions (thus achieving the superior harmonic performance of OPPs).

Chapter 6 expands the formulation of the small-signal controller to impose bounds on the state vector in the form of constraints. First, the bounds are formulated as linear constraints. Additional decision variables are included to relax the bounds (resulting in so-called soft constraints) in order to maintain a feasible optimization problem. Thereafter, the constraints are added to the optimization problem underlying the small-signal controller. Simulation results confirm that converter states are kept within their respective bounds, although some relaxation of the bounds is present.

Chapter 7 integrates balancing of the neutral-point potential in the small-signal controller. First, a method is derived that determines the steady-state trajectory, that includes the effect of the neutral-point potential, of an OPP-modulated converter system. An efficient method to model the modifications to the absolute value of a pulse pattern is shown. Finally, the small-signal controller with integrated balancing of the neutral-point potential is derived. Simulation results demonstrate that the controller balances the neutral-point potential very effectively, even under operating conditions where traditional methods are insufficient.

Chapter 8 discusses the implementation of the control algorithm. Specifically, the aspects of the control algorithm with a high computational burden are analyzed, and recommendations are given to reduce the computations and efficiently implement these aspects. A brief summary of the control algorithm that is implemented on an FPGA is given; this includes an overview of some of the design choices, resource usage, and execution time. A hardware-in-the-loop simulation demonstrates that the control algorithm can execute in real-time, within a short sampling interval of 25 µs, on a low-cost FPGA.

Part III: Summary and Outlook summarizes the main results of the thesis. Recommendations for future work are given.

Chapter 2

Mathematical Optimization

Finding the minimum of a constrained convex quadratic program by using the gradient projection method is a crucial step for the control algorithm developed in this thesis. To some readers, these terms and concepts might be unfamiliar. The intent of the following chapter is to present a basic introduction to optimization that will be sufficient for this thesis. Proofs are omitted and explanations are kept relatively simple. Readers who are interested in a more in-depth (and complete) understanding of optimization are referred to any of the following classical textbooks [6, 7, 8, 9, 10]. For an accessible introduction, see [11].

Chapter Contents

2.1 General Optimization Problems

A general nonlinear optimization problem can be stated as

$$\min_{\boldsymbol{z}} \quad f(\boldsymbol{z}) \tag{2.1a}$$

$$\text{subject to} \quad g_i(\boldsymbol{z}) \leq 0 \quad \text{for } i = 1, 2, \ldots, n_g \tag{2.1b}$$

$$h_i(\boldsymbol{z}) = 0 \quad \text{for } i = 1, 2, \ldots, n_h, \tag{2.1c}$$

where $\boldsymbol{z} \in \mathcal{Z} \subseteq \mathbb{R}^{n_z}$ is referred to as the *decision variable* (or optimization variable). The continuous function $f(\boldsymbol{z}) : \mathbb{R}^{n_z} \to \mathbb{R}$ with the argument $\boldsymbol{z}$ is known as the *objective function* (or cost function). A value of $f(\boldsymbol{z})$ will be referred to as *cost*. The functions $g_i(\boldsymbol{z}) : \mathbb{R}^{n_z} \to \mathbb{R}$ and $h_i(\boldsymbol{z}) : \mathbb{R}^{n_z} \to \mathbb{R}$ are referred to as the *inequality* and *equality constraint functions*, respectively. Usually, the constraint functions are simply referred to as the *constraints*. The constraints define a subset $\mathcal{Z} = \{\boldsymbol{z} : g_i(\boldsymbol{z}) \leq 0, h_i(\boldsymbol{z}) = 0\}$ on $\mathbb{R}^{n_z}$ to which the decision variable must belong to when $f(\boldsymbol{z})$ is minimized. The set $\mathcal{Z}$ is referred to as the *feasible region*.

A point $\boldsymbol{z}_{\text{min}}$ is called a *local minimum* if it has the lowest cost in a neighbourhood close to $\boldsymbol{z}_{\text{min}}$,

$$f(\boldsymbol{z}_{\text{min}}) \leq f(\boldsymbol{z}) \quad \text{for any } \boldsymbol{z} \text{ in a neighbourhood of } \boldsymbol{z}_{\text{min}}.$$

Furthermore, a point $\boldsymbol{z}_{\text{opt}}$ is referred to as a *global minimum* if it has the lowest cost over the entire feasible region,

$$f(\boldsymbol{z}_{\text{opt}}) \leq f(\boldsymbol{z}) \quad \text{for all } \boldsymbol{z} \in \mathcal{Z}.$$

This point is referred to as the *optimal solution* (or global minimizer). In the case that $\boldsymbol{z}_{\text{opt}}$ is unique, it is referred to as a *strict global minimum* and implies

$$f(\boldsymbol{z}_{\text{opt}}) < f(\boldsymbol{z}) \quad \text{for all } \boldsymbol{z} \in \mathcal{Z}, \boldsymbol{z} \neq \boldsymbol{z}_{\text{opt}}.$$

The value at which the (global) minimum is obtained is denoted by $f_{\text{opt}} = f(\boldsymbol{z}_{\text{opt}}) = \min_{\boldsymbol{z} \in \mathcal{Z}} f(\boldsymbol{z})$. Often it will be stated

$$\boldsymbol{z}_{\text{opt}} = \arg\min_{\boldsymbol{z} \in \mathcal{Z}} f(\boldsymbol{z}),$$

which indicates that the argument (or solution) that minimizes $f(\boldsymbol{z})$ is returned.

In general, (2.1) can not be solved algebraically; a numerical method (an algorithm) must be employed to solve the problem. A few of these methods will be discussed in Section 2.3.

It is important to be able to recognize when a minimum has been achieved during minimization. For simplicity, consider the case where there are no constraints, implying $\mathcal{Z} = \mathbb{R}^{n_z}$; this is commonly referred to as *unconstrained optimization*. The following conditions that will be presented are known as *optimality conditions*. For a more comprehensive analysis (with proofs), refer to [6, Section 1.1]. Recall from calculus that a minimum is a stationary point, meaning

$$\nabla f(\boldsymbol{z}_{\text{min}}) = \begin{bmatrix} \frac{\partial f(\boldsymbol{z}_{\text{min}})}{\partial z_1} & \frac{\partial f(\boldsymbol{z}_{\text{min}})}{\partial z_2} & \cdots & \frac{\partial f(\boldsymbol{z}_{\text{min}})}{\partial z_{n_z}} \end{bmatrix}^{\text{T}} = \boldsymbol{0}_{n_z}, \tag{2.2}$$

where $\nabla f(\boldsymbol{z}) \in \mathbb{R}^{n_z}$ (a vector of first-order partial derivatives) is known as *the gradient*. This is called the *necessary first-order optimality condition*. However, the gradient is also zero at maxima and saddle points; more information is thus required to identify a minimum. The matrix of second-order partial derivatives

$$\boldsymbol{H}(\boldsymbol{z}) = \nabla^2 f(\boldsymbol{z}) \in \mathbb{R}^{n_z \times n_z}, \tag{2.3}$$

which is referred to as the *Hessian matrix* (or simply the Hessian), supplies the required information. Its (i,j)th entry is given by $\boldsymbol{H}_{(i,j)}(\boldsymbol{z}) = \frac{\partial^2 f(\boldsymbol{z})}{\partial z_i z_j}$. Note that the Hessian is symmetrical. At a minimum, the Hessian $\boldsymbol{H}(\boldsymbol{z}_{\min})$ is positive semidefinite[1], that is,

$$(\boldsymbol{z} - \boldsymbol{z}_{\min})^{\mathrm{T}} \boldsymbol{H}(\boldsymbol{z}_{\min})(\boldsymbol{z} - \boldsymbol{z}_{\min}) \geq 0 \quad \text{for any } \boldsymbol{z} \text{ in a neighbourhood of } \boldsymbol{z}_{\min}.$$

This is called the *necessary second-order optimality condition*. Note that these two optimality conditions are called the *necessary* conditions; they are not always sufficient and a maximum or saddle point *can also* satisfy these conditions.[2] A stricter condition that guarantees a minimum has been obtained is when the Hessian $\boldsymbol{H}(\boldsymbol{z}_{\min})$ is positive definite,

$$(\boldsymbol{z} - \boldsymbol{z}_{\min})^{\mathrm{T}} \boldsymbol{H}(\boldsymbol{z}_{\min})(\boldsymbol{z} - \boldsymbol{z}_{\min}) > 0 \quad \text{for any } \boldsymbol{z} \text{ in a neighbourhood of } \boldsymbol{z}_{\min}.$$

This is called the *sufficient second-order optimality condition*.

For constrained optimization there are similar optimality conditions, which are known as the *Karush-Kuhn-Tucker conditions*. Interested readers are referred to [6, Section 4.3.1] for more information.

Thus far, no assumptions have been made regarding the convexity of the optimization problem. A special and important class of optimization problems are *convex problems*. In order for a problem to be convex, the Hessian $\boldsymbol{H}(\boldsymbol{z})$ of the objective function must be positive semidefinite for all $\boldsymbol{z} \in \mathcal{Z}$ and $\mathcal{Z}$ must be a convex set [6, Proposition B.4]. A set $\mathcal{Z}$ is defined as convex if [6, Appendix B.1]

$$\ell \boldsymbol{z}_1 + (1-\ell)\boldsymbol{z}_2 \in \mathcal{Z} \quad \text{for all } \boldsymbol{z}_1, \boldsymbol{z}_2 \in \mathcal{Z} \text{ and for all } \ell \in [0,1],$$

which can be interpreted as the line segment that joins any two points in a set must also be contained in the set itself. An important and useful characteristic of convex problems is that any stationary point [see (2.2)] is a minimum, meaning only the first-order optimality condition is required. In fact, any minimum is a global minimum. It is thus easy to determine once the optimal solution is found, and these problems may be efficiently solved with many optimization algorithms. In contrast, nonconvex optimization problems can have multiple minima, maxima, and saddle points, and thus require significant effort in order to find the global minimum. In the sequel, unless explicitly mentioned otherwise, only convex problems are considered. A particular type of convex optimization problem, that is encountered frequently throughout this thesis, is considered in the next section.

2.2 Quadratic Programs

A type of convex optimization problem that arises frequently in many fields, one of them being optimal control, is a *quadratic program* (QP), which can be stated as

$$\begin{aligned} \min_{\boldsymbol{z}} \quad & \tfrac{1}{2}\boldsymbol{z}^{\mathrm{T}}\boldsymbol{H}\boldsymbol{z} + \boldsymbol{c}^{\mathrm{T}}\boldsymbol{z} && \text{(2.4a)} \\ \text{subject to} \quad & \boldsymbol{A}\boldsymbol{z} \leq \boldsymbol{b} && \text{(2.4b)} \\ & \boldsymbol{A}_{\mathrm{eq}}\boldsymbol{z} = \boldsymbol{b}_{\mathrm{eq}}, && \text{(2.4c)} \end{aligned}$$

[1] A matrix $\boldsymbol{M}$ is called *positive semidefinite* if $\boldsymbol{z}^{\mathrm{T}}\boldsymbol{M}\boldsymbol{z} \geq 0$ for all $\boldsymbol{z}$, and *positive definite* if $\boldsymbol{z}^{\mathrm{T}}\boldsymbol{M}\boldsymbol{z} > 0$ for all nonzero $\boldsymbol{z}$.

[2] For example, consider $f(z) = z^3$, which has an inflection point at $z = 0$ but satisfies the necessary optimality conditions at that point [$\frac{\mathrm{d}f(0)}{\mathrm{d}z} = 0$ and $\frac{\mathrm{d}^2 f(0)}{\mathrm{d}z^2} = 0$].

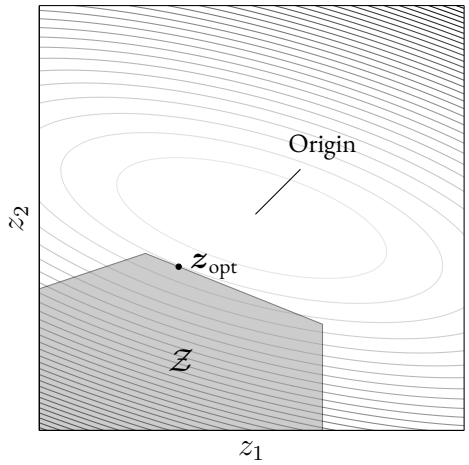


Figure 2.1: Illustration of an inequality-constrained QP, where $\mathcal{Z} = \{\boldsymbol{z} : \boldsymbol{A}\boldsymbol{z} \leq \boldsymbol{b}\}$. The contour lines indicate the level sets, and those that are closer to the origin have a lower cost. The optimal solution $\boldsymbol{z}_{\text{opt}}$ is the point in the set $\mathcal{Z}$ with the lowest cost. The shape of the contour lines are defined by the Hessian.

where the Hessian $\boldsymbol{H}$ is constant and assumed to be positive (semi)definite[3], and $\boldsymbol{c} \in \mathbb{R}^{n_z}$ a vector. Note that the inequality in (2.4b) applies to each component. A QP can have linear equality and inequality constraints, which can be written in compact matrix notation (a system of linear equations) using $\boldsymbol{A} \in \mathbb{R}^{n_g \times n_z}$, $\boldsymbol{b} \in \mathbb{R}^{n_g}$, $\boldsymbol{A}_{\text{eq}} \in \mathbb{R}^{n_h \times n_z}$, and $\boldsymbol{b}_{\text{eq}} \in \mathbb{R}^{n_h}$, as shown in (2.4b) and (2.4c). These linear constraints are convex and form a *polyhedron*. A geometric illustration of a QP is shown in Figure 2.1.

Since the gradient of a quadratic function is

$$\nabla f(\boldsymbol{z}) = \boldsymbol{H}\boldsymbol{z} + \boldsymbol{c},$$

it can easily be seen from (2.2) that the solution of an *unconstrained* QP can be obtained by solving

$$\boldsymbol{H}\boldsymbol{z}_{\text{opt}} = -\boldsymbol{c}.$$

If the Hessian $\boldsymbol{H}$ is only positive semidefinite (and therefore singular), the optimal solution $\boldsymbol{z}_{\text{opt}}$ is not unique. In the case that the Hessian $\boldsymbol{H}$ is positive definite, there will be a unique optimal solution given by

$$\boldsymbol{z}_{\text{opt}} = -\boldsymbol{H}^{-1}\boldsymbol{c}.$$

This is sometimes referred to as a *strictly convex* problem.

2.3 Optimization Algorithms

In order to solve the QP of (2.4), or in general (2.1), an optimization algorithm must be employed, which is an iterative method. These algorithms can be split into three categories [10, Section 1.1.2]: zero-order methods that only evaluate the objective function, first-order methods that evaluate the objective function and its gradient, and second-order methods that evaluate the objective function, its gradient, and the Hessian. Two complexity measures can be associated with an algorithm [10, Section 1.1.2]. The first is the *analytical complexity*, which refers to the number of iterations an algorithm requires to solve the problem accurately (in other words, how quickly the algorithm

[3] In general when referring to a QP, it is assumed to be convex. However, if the Hessian is indefinite, the QP will be nonconvex and difficult to solve globally [12].

converges). The second measure is called the *arithmetic complexity*, which refers to the arithmetic operations per iteration of an algorithm (that is, the computational burden). The higher the order of the method, the faster the convergence rate and the higher the computational burden (and vice versa), implying a trade-off between analytical and arithmetic complexity. Both of these complexity measures are of particular concern when an algorithm has to solve a problem with sufficient accuracy in real-time (typically within a few microseconds) on a hardware-constrained device. This is the case with a model predictive controller executing in real-time on an embedded system, which is discussed in Chapter 8.

An iteration of an algorithm usually has the form

$$\boldsymbol{z}_{k+1} = \boldsymbol{z}_k + s_k \Delta \boldsymbol{z}_k, \tag{2.5}$$

where $\boldsymbol{z}_k$ and $\boldsymbol{z}_{k+1}$ are the current and next iteration, respectively; and $\Delta \boldsymbol{z}_k \in \mathbb{R}^{n_z}$ and $s_k \in \mathbb{R}$ (which are determined by the algorithm) are the direction of the step and the stepsize, respectively.

2.3.1 Newton's Method

First consider one of the most advanced and fastest unconstrained optimization algorithms, *Newton's method*, which is a second-order method. This method is a well-known root-finding algorithm, and is used in many fields outside of optimization. For (unconstrained) optimization problems, Newton's method attempts to find the roots of the gradient; it attempts to solve (2.2) iteratively. The method is shown in Algorithm 1. Newton's method exhibits quadratic convergence[4], which is extremely fast. In fact, Newton's method (with a full step) can solve an (unconstrained) QP within a single iteration.

Algorithm 1 Newton's Method for Unconstrained Optimization

1: **procedure** NEWTONSMETHOD($f(\boldsymbol{z})$, $\boldsymbol{z}_0$) ▷ $\boldsymbol{z}_0$ is the initial iterate
2: **while** stopping criterion is not met **do**
3: Solve $\boldsymbol{H}(\boldsymbol{z}_k)\Delta \boldsymbol{z}_k = -\nabla f(\boldsymbol{z}_k)$ ▷ Find the step direction $\Delta \boldsymbol{z}_k$
4: Find the stepsize $s_k \in (0, 1]$ so that $f(\boldsymbol{z}_k)$ decreases sufficiently to the next iterate
5: $\boldsymbol{z}_{k+1} \leftarrow \boldsymbol{z}_k + s_k \Delta \boldsymbol{z}_k$
6: **end while**
7: **return** $\boldsymbol{z}_{\text{opt}}$ ▷ $\boldsymbol{z}_{\text{opt}}$ is the final iterate
8: **end procedure**

Regarding constrained optimization algorithms, two popular methods available to solve (2.4) with relatively few iterations are *active-set* methods [7, Section 16.5] and *interior-point* methods [13] (both make use of Newton's method). In particular, interior-point methods are some of the most commonly used algorithms for general nonlinear optimization, whereas active-set methods are typically restricted to linearly-constrained QPs[5] (but are well suited for such problems). Although these algorithms have fast convergence[6] (good analytical complexity), they have significant arithmetic complexity. At *every* iteration these algorithms require solving a system of linear equations (similar to Newton's method, see line 3 in Algorithm 1), which is a nontrivial task. Solving a system

[4] To be more specific, quadratic convergence is achieved close to a minimum. See [6, Section 1.6], [7, Section 3.4], and [9, Section 9.5.2] for more information on Newton's method.

[5] For an active-set method that can be used for general nonlinear programs, refer to *sequential quadratic programming* [7, Chapter 18].

[6] The rate of convergence of active-set methods are unknown [14, Section 7.3.2], but in practice they typically perform well for small- to medium-sized problems [7, Section 16.5].

of linear equations involves a considerable amount of arithmetic operations (growing cubically with the problem size), including divisions. Special attention is required to ensure the solver is robust (numerically stable). Unless the problem has some structure that can be exploited, see [14, Section 7.3.2] for some examples, active-set and interior-point methods will be difficult to implement on embedded hardware. In the next section, a method with a very low computational burden, that is well suited to solve (2.4) in real-time, is reviewed.

2.3.2 The Gradient Method

The *gradient method* (also referred to as the steepest descent method) is a first-order optimization algorithm. A property of the gradient $\nabla f(\boldsymbol{z})$ is that it points towards the steepest ascent at a given point. The gradient method, pragmatically, takes a step in the opposite direction of the gradient, towards the steepest descent. Thus, in view of (2.5),

$$\Delta \boldsymbol{z}_k = -\nabla f(\boldsymbol{z}_k). \tag{2.6}$$

There are a few stepsize rules for s_k of (2.5). A very simple and easily implementable rule is a *fixed stepsize*. However, the stepsize (which must be chosen in advance) must not be too large so that the algorithm diverges, nor too small so that convergence is extremely slow. Choosing the optimal (fixed) stepsize is discussed soon. For other stepsize rules, see [6, Section 1.2.1] and [10, Section 1.2.3].

Before continuing to explain the gradient method, it is worthwhile to motivate the use of the method. The algorithm is simple yet robust, and is well suited to solve the optimization problems underlying real-time model predictive controllers [14, Section 7.4]. The gradient method can tolerate rounding errors [15], which is ideal when using fixed-point arithmetic. In some ways, the gradient method is the opposite of Newton's method; where Newton's method has fast convergence at the expense of the computational burden, the gradient method is a (very) simple algorithm at the expense of the convergence rate (at best, linear convergence [10, Section 2.1.5]). However, it is easy to motivate that algorithms yielding a solution with floating-point accuracy are not required for real-time controllers in practical conditions. The solution should simply have sufficient accuracy, which for power electronic applications is typically not that tight.

There are also other interesting gradient methods, such as *Nesterov's fast gradient method* [16], but these methods are not discussed for the sake of brevity.

The Unconstrained Gradient Method

As mentioned previously, a fixed stepsize is used. In order to choose a (fixed) stepsize in advance that guarantees convergence [6, Proposition 1.2.2], *Lipschitz continuity* of the gradient is required, which is defined as

$$\|\nabla f(\boldsymbol{z}_2) - \nabla f(\boldsymbol{z}_1)\|_2 \leq L_c \|\boldsymbol{z}_2 - \boldsymbol{z}_1\|_2 \quad \text{for all } \boldsymbol{z}_1, \boldsymbol{z}_2 \in \mathbb{R}^{n_z},$$

where $\|\boldsymbol{\xi}\|_2$ is the 2-norm (or Euclidean norm) of the vector $\boldsymbol{\xi}$, and L_c is a strictly positive *Lipschitz constant*. If a function is twice continuously differentiable, then a Lipschitz constant can be characterized by the Hessian as [10, Lemma 1.2.2]

$$\|\boldsymbol{H}(\boldsymbol{z})\|_2 \leq L_c,$$

where $\|\boldsymbol{M}\|_2$ is the (induced) 2-norm of the matrix $\boldsymbol{M}$. Thus, for a quadratic function,

$$L_c = \|\boldsymbol{H}\|_2, \tag{2.7}$$

which is referred to as the *tight* Lipschitz constant. Note that L_c is the largest eigenvalue of the Hessian [17, Section 11.2]. Furthermore, denote with μ the smallest eigenvalue of the Hessian,

$$\mu = \frac{1}{\|\boldsymbol{H}^{-1}\|_2}, \tag{2.8}$$

which is referred to as the *convexity parameter* [18, Section 5.2].

In order for the gradient method to converge, the stepsize must satisfy [10, Theorem 2.1.14]

$$s \in \left(0, \frac{2}{L_c}\right). \tag{2.9}$$

If the convexity parameter μ is taken into account, the optimal stepsize (resulting in the fastest convergence) is $s = \frac{2}{L_c+\mu}$ [10, Section 2.1.5]. However, calculating the smallest eigenvalue in real-time conditions is highly demanding. Whereas the largest eigenvalue can be overestimated with little computation (instead of evaluating the 2-norm), it is not trivial to get a nonzero lower bound on the smallest eigenvalue. This is further discussed in Section 8.2.1. If the convexity parameter is not taken into account, the ideal stepsize, albeit with slower convergence, is $s = \frac{1}{L_c}$ [10, Section 2.1.5]. Note that if the stepsize is $s \geq \frac{2}{L_c}$, the gradient method will not converge, implying that the (tight) Lipschitz constant L_c must not be underestimated by a factor of two or more. Algorithm 2 presents the gradient method with a stepsize $s = \frac{1}{L_c}$. Figure 2.2 shows the first ten iterations of the gradient method when considering different (fixed) stepsizes.

Algorithm 2 Gradient Method for Unconstrained Optimization

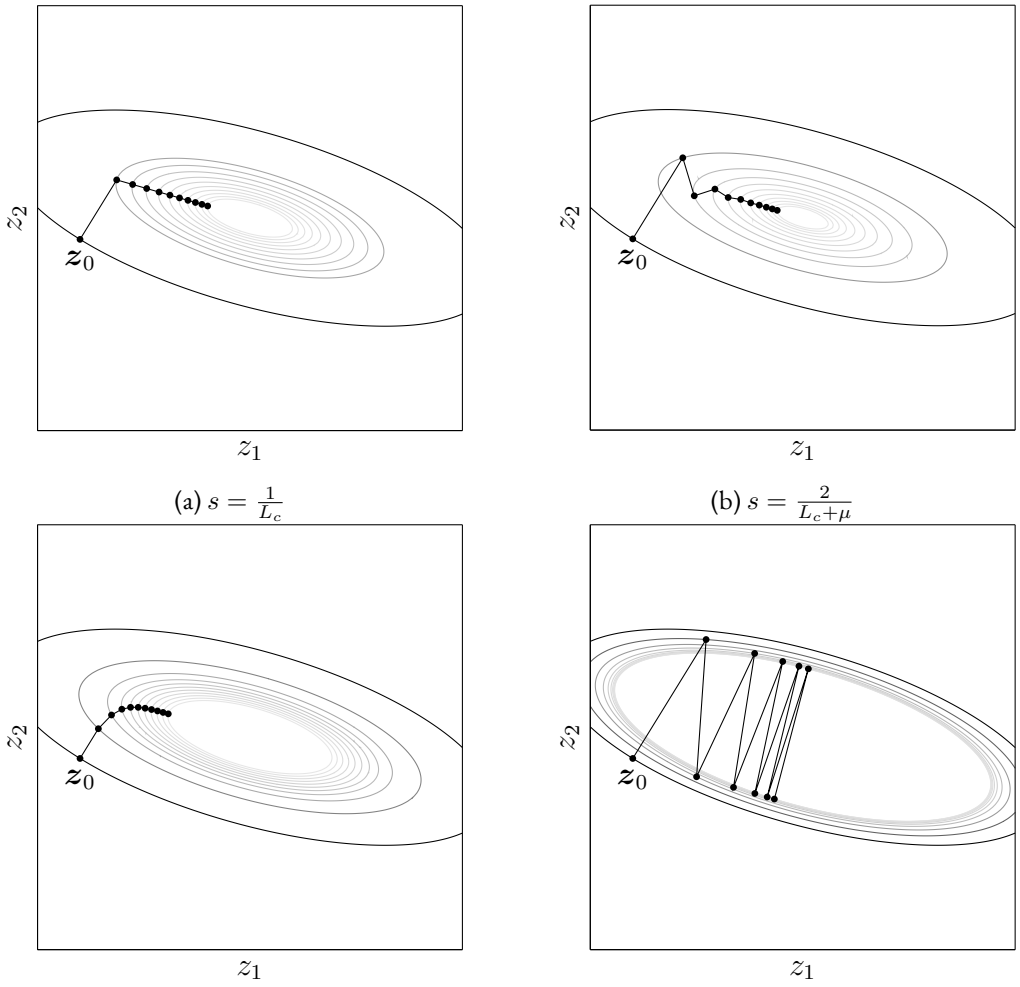
1: **procedure** GRADIENTMETHOD($f(\boldsymbol{z})$, $\boldsymbol{z}_0$) ▷ $\boldsymbol{z}_0$ is the initial iterate
2: **while** stopping criterion is not met **do**
3: $\boldsymbol{z}_{k+1} \leftarrow \boldsymbol{z}_k - \frac{1}{L_c}\nabla f(\boldsymbol{z}_k)$
4: **end while**
5: **return** $\boldsymbol{z}_{\text{opt}}$ ▷ $\boldsymbol{z}_{\text{opt}}$ is the final iterate
6: **end procedure**

The conditioning of the Hessian has a significant impact on the convergence of the gradient method [6, Section 1.3.2]. For a symmetrical positive definite matrix, the so-called *conditioning number* is [17, Section 11.2]

$$\kappa = \frac{L_c}{\mu} \geq 1. \tag{2.10}$$

Matrices with a high conditioning number are referred to as being *ill-conditioned* and result in a slow convergence of the gradient method, whereas matrices with a low conditioning number are *well-conditioned* and converge quickly. Figure 2.3 illustrates this. In general, there is no specific conditioning number where a matrix is suddenly considered to be ill-conditioned, as the numerical accuracy of software and the data also determine how the conditioning number affects a problem. However, in view of the gradient method, the number of iterations required for a specific accuracy is proportional to the conditioning number. In fact, if the Hessian is singular (which has a conditioning number of infinity), the gradient method will not converge. Since the gradient method is extremely reliant on conditioning, it is not considered a general purpose method and is typically absent in optimization suites.

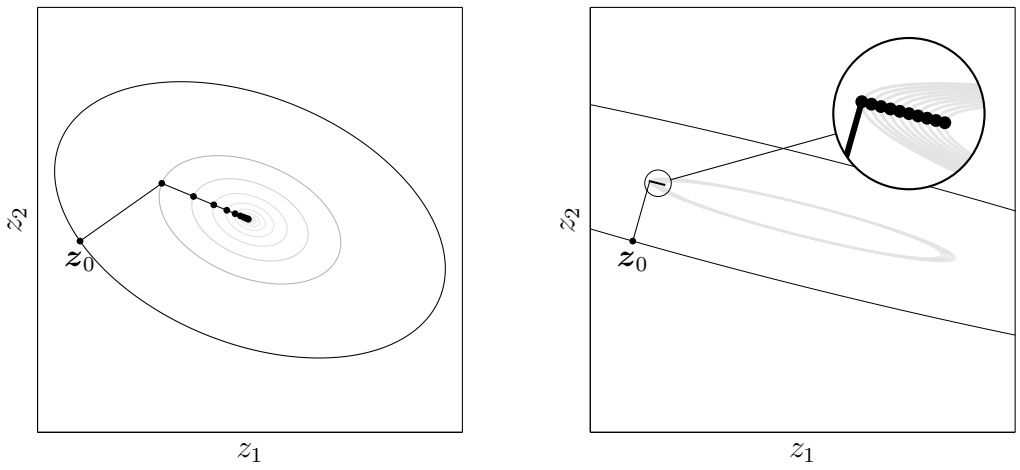
For further reading on gradient methods, refer to [6, Sections 1.2 and 1.3] and [10, Sections 1.2.3, 2.1.5, 2.2.2, and 2.2.2]. For an accessible source, see [11, Chapter 4]. For further reading on first-order methods, see [18].



(a) $s = \frac{1}{L_c}$ (b) $s = \frac{2}{L_c+\mu}$

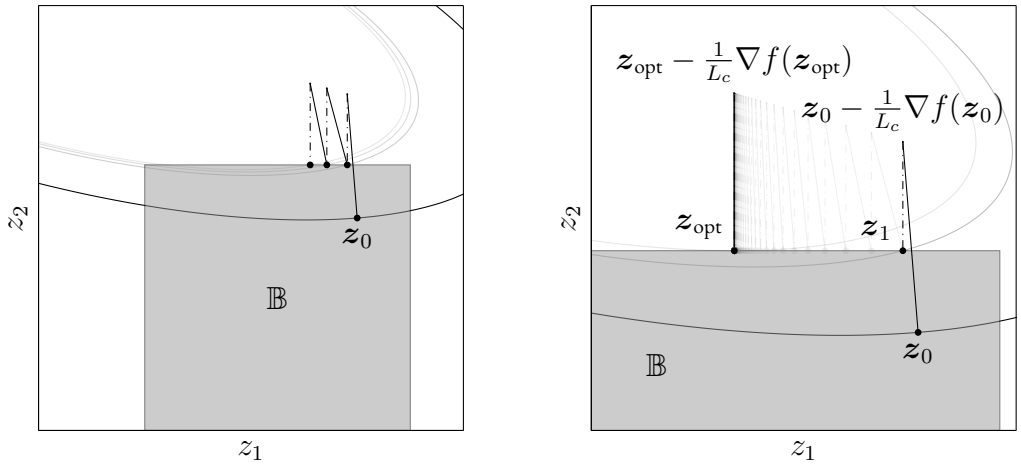
(c) $s = \frac{1}{2L_c}$ (d) $s = \frac{2}{L_c}$

Figure 2.2: The first ten iterations of the gradient method, illustrating the convergence of different stepsizes. It can be observed that the gradient is orthogonal to the contour line at a given iteration.



(a) Well-conditioned Hessian, with $\kappa = 2.78$. (b) Ill-conditioned Hessian, with $\kappa = 90.4$.

Figure 2.3: The impact of conditioning on the convergence of the gradient method. A well-conditioned Hessian is characterized by circular contour lines, whereas an ill-conditioned Hessian has elongated contour lines.



(a) First three iterations.

(b) The behaviour of gradient method at the first iteration and at the optimal solution.

Figure 2.4: The gradient projection method with box constraints.

The Gradient Projection Method

The gradient method can be extended so that it is able to minimize a function over a convex set $\mathcal{Z}$. First introduce the (orthogonal) *projection operator* [6, Proposition 1.1.4],

$$\pi_{\mathcal{Z}}(\boldsymbol{x}) = \arg\min_{\boldsymbol{z}\in\mathcal{Z}} \tfrac{1}{2}\|\boldsymbol{z}-\boldsymbol{x}\|_2^2, \tag{2.11}$$

which projects $\boldsymbol{x}$ onto the set $\mathcal{Z}$. Then, after the (unconstrained) gradient method takes a step, the result is projected onto the set $\mathcal{Z}$ by using (2.11). This is referred to as the *gradient projection method*, and is described in Algorithm 3.

Algorithm 3 Gradient Projection Method for Constrained Optimization

procedure GRADIENTPROJECTIONMETHOD($f(\boldsymbol{z})$, $\boldsymbol{z}_0$) ▷ $\boldsymbol{z}_0$ is the initial iterate
  while stopping criterion is not met **do**
    $\boldsymbol{z}_{k+1} \leftarrow \pi_{\mathcal{Z}}\left(\boldsymbol{z}_k - \frac{1}{L_c}\nabla f(\boldsymbol{z}_k)\right)$ ▷ Take step and project result onto $\mathcal{Z}$
  end while
  return $\boldsymbol{z}_{\text{opt}}$ ▷ $\boldsymbol{z}_{\text{opt}}$ is the final iterate
end procedure

In order for the method to be practically viable, the projection operation should be relatively simple. For some sets, a simple closed-form solution exists. As an example, consider so-called *box constraints* (or bounds),

$$\mathbb{B} = \{\boldsymbol{z} : \underline{z}_i \le z_i \le \bar{z}_i \text{ for } i = 1, 2, \ldots, n_z\},$$

where $\underline{z}_i$ and $\bar{z}_i$ are the lower and upper bounds, respectively, of the ith component of $\boldsymbol{z}$. The projection of the ith component is defined as

$$\begin{aligned}[\pi_{\mathbb{B}}(\boldsymbol{x})]_i &= \begin{cases} \underline{z}_i & \text{if } x_i < \underline{z}_i \\ \bar{z}_i & \text{if } x_i > \bar{z}_i \\ x_i & \text{else} \end{cases} \\ &= \min\{\max\{x_i, \underline{z}_i\}, \bar{z}_i\}.\end{aligned}$$

An illustration of the gradient projection method for a QP with box constraints is shown in Figure 2.4. For a list of projection rules for other sets, see [18, Table 6.1] and [14, Table 5.1].

In general, a closed-form solution is not available for the projection. Consider a polyhedron defined by n_g inequality constraints, $\mathcal{Z} = \{\boldsymbol{z} : \boldsymbol{A}\boldsymbol{z} \leq \boldsymbol{b}\}$, where $\boldsymbol{A} \in \mathbb{R}^{n_g \times n_z}$ and $\boldsymbol{b} \in \mathbb{R}^{n_g}$. An *approximate* projection can be obtained by solving (2.11) with an algorithm. Refer to (2.11) as the *primal* problem. By using the notion of *duality*, it can be shown that the (approximate) projection also follows as

$$\pi_{\mathcal{Z}}(\boldsymbol{x}) = \boldsymbol{x} - \boldsymbol{A}^{\mathrm{T}}\boldsymbol{\eta}_{\mathrm{opt}} \tag{2.12}$$

with

$$\boldsymbol{\eta}_{\mathrm{opt}} = \arg\min_{\boldsymbol{\eta} \geq \boldsymbol{0}} \tfrac{1}{2}\boldsymbol{\eta}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{\eta} + (\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x})^{\mathrm{T}}\boldsymbol{\eta}, \tag{2.13}$$

where $\boldsymbol{\eta} \in \mathbb{R}^{n_g}$ is known as the *Lagrange multiplier*. The derivation of (2.12) can be found in Appendix A. The problem (2.13), which is also a QP, is referred to as the *dual* problem. Note that the constraints of the dual problem are extremely simple; $\boldsymbol{\eta}$ should be nonnegative. The projection onto the set of nonnegative real numbers $\mathbb{R}_+^{n_g}$ is

$$\pi_{\mathbb{R}_+^{n_g}}(\boldsymbol{\eta}) = \max\{\boldsymbol{0}_{n_g}, \boldsymbol{\eta}\},$$

where the max operation is componentwise (the ith operation is $\max\{0, \eta_i\}$). Thus, the dual problem itself can be solved with the gradient projection method, with the kth iteration being

$$\boldsymbol{\eta}_{k+1} = \max\left\{\boldsymbol{0}_{n_g}, \boldsymbol{\eta}_k - s\left(\boldsymbol{A}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{\eta}_k + (\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x})\right)\right\}, \tag{2.14}$$

where the stepsize is $s = \frac{1}{L_d}$ or, if possible, $s = \frac{2}{\mu_d + L_d}$. In this case, the Lipschitz constant L_d and convexity parameter μ_d refer to the Hessian of the dual projection problem, $\boldsymbol{A}\boldsymbol{A}^{\mathrm{T}}$. The accuracy of the projection relies on how accurately (2.13) is solved: more iterations of (2.14) result in a more accurate projection. Obviously, the conditioning of $\boldsymbol{A}\boldsymbol{A}^{\mathrm{T}}$ is also crucial as to how many iterations are required for an accurate projection. Once the dual variable $\boldsymbol{\eta}$ is calculated (up to a certain accuracy), (2.12) is used to retrieve the primal solution and complete the (approximate) projection. The approximate projection plays a central role in Section 8.3.2, and is discussed further there.

For further reading on the gradient projection method, refer to [6, Section 3.3], [10, Section 2.2.5], and [11, Section 9.4].

Chapter 3

Power Electronics and Optimized Pulse Patterns

A medium-voltage grid-connected neutral-point-clamped converter that is modulated by optimized pulse patterns is used as the primary case study throughout this thesis. This chapter first explains a few preliminary concepts regarding power electronics and power systems. A review of the neutral-point-clamped converter, which is one of the most popular converter topologies for medium-voltage applications, is then given. Thereafter, the application of a converter connected to the grid is discussed. This involves (briefly) reviewing the harmonic standards that a grid-connected converter must satisfy and how the system can be modelled. The chapter concludes with optimized pulse patterns which, at the low switching frequencies that medium-voltage converter systems operate, is arguably the modulation technique with the most benefits.

Chapter Contents

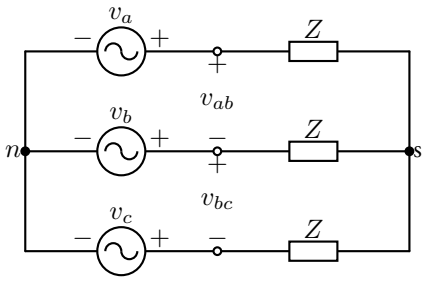


Figure 3.1: A three-phase voltage source connected to a load.

3.1 Preliminaries

The following section explains some useful concepts in power electronics. In particular, the Clarke transformation and the per-unit system. In order to give some context to the aforementioned concepts, three-phase systems are first reviewed. The majority of the content in this section is adapted from [4, Section 2.1].

3.1.1 Three-Phase Systems

A general three-phase system is shown in Figure 3.1. It is assumed that the load Z is balanced in all three phases. The point n denotes the neutral point of the three-phase source, and the point s is the star point of the load. Only star-connected systems are considered. Typically, the star point s is floating. In the case that the star point s is connected to the neutral point n, the system turns into three decoupled single-phase systems, and the benefits of a three-phase system are lost.

The three-phase voltage source is assumed to be balanced with a positive phase sequence, with the line-to-neutral (phase) voltages being

$$\begin{aligned} v_a(t) &= \sqrt{2}V_p \sin(\omega_1 t) \\ v_b(t) &= \sqrt{2}V_p \sin(\omega_1 t - \frac{2\pi}{3}) \\ v_c(t) &= \sqrt{2}V_p \sin(\omega_1 t + \frac{2\pi}{3}), \end{aligned}$$

where V_p is the root-mean-square (rms) voltage and $\omega_1 = 2\pi f_1$ is the fundamental angular frequency. Unless mentioned otherwise, all voltages and currents are rms quantities. The line-to-line voltages are

$$\begin{aligned} v_{ab}(t) = v_a(t) - v_b(t) &= \sqrt{2}V_{\mathrm{LL}} \sin(\omega_1 t + \frac{\pi}{6}) \\ v_{bc}(t) = v_b(t) - v_c(t) &= \sqrt{2}V_{\mathrm{LL}} \sin(\omega_1 t - \frac{\pi}{2}) \\ v_{ca}(t) = v_c(t) - v_a(t) &= \sqrt{2}V_{\mathrm{LL}} \sin(\omega_1 t + \frac{5\pi}{6}), \end{aligned}$$

where $V_{\mathrm{LL}} = \sqrt{3}V_p$.

The real, reactive, and apparent power of a three-phase system are

$$\begin{aligned} P &= 3V_p I_p \cos(\phi) \\ Q &= 3V_p I_p \sin(\phi) \\ S &= 3V_p I_p, \end{aligned}$$

respectively. Here, I_p is the (rms) phase current and ϕ is the angle between the voltage and current. The power factor is defined as

$$\text{PF} = |\cos(\phi)| = \left|\frac{P}{S}\right|.$$

If ϕ is positive, the current is lagging the voltage and signifies a lagging power factor, and vice versa when ϕ is negative. Unity power factor is achieved when ϕ is zero.

3.1.2 Clarke Transformation

It is common to make use of transformations when modelling power electronic systems. One such transformation is the *Clark transformation*, which maps the three-phase abc coordinate system to the (orthogonal) $\alpha\beta 0$ coordinate system. The terms $\alpha\beta 0$ plane, $\alpha\beta 0$ reference frame, and stationary orthogonal coordinate system are used interchangeably. The transformation is defined as [19]

$$\boldsymbol{\xi}_{\alpha\beta 0} = \boldsymbol{K}\boldsymbol{\xi}_{abc},$$

where $\boldsymbol{\xi}_{\alpha\beta 0} = [\xi_\alpha\ \xi_\beta\ \xi_0]^{\mathrm{T}}$ and $\boldsymbol{\xi}_{abc} = [\xi_a\ \xi_b\ \xi_c]^{\mathrm{T}}$, with the transformation matrix

$$\boldsymbol{K} = \frac{2}{3}\begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \\ \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \end{bmatrix}.$$

Accordingly, the inverse transformation is

$$\boldsymbol{\xi}_{abc} = \boldsymbol{K}^{-1}\boldsymbol{\xi}_{\alpha\beta 0}$$

with the inverse transformation matrix

$$\boldsymbol{K}^{-1} = \begin{bmatrix} 1 & 0 & 1 \\ -\frac{1}{2} & \frac{\sqrt{3}}{2} & 1 \\ -\frac{1}{2} & -\frac{\sqrt{3}}{2} & 1 \end{bmatrix}.$$

The 0-component, which is the *common-mode* term, is often ignored. This is always the case in this thesis, as common-mode terms are either zero or do not contribute to phase current (this is further explained in Section 3.2.1). The reduced Clarke transformation and its inverse are introduced as

$$\boldsymbol{\xi}_{\alpha\beta} = \boldsymbol{K}\boldsymbol{\xi}_{abc} \tag{3.4}$$

and

$$\boldsymbol{\xi}_{abc} = \boldsymbol{K}^{-1}\boldsymbol{\xi}_{\alpha\beta}, \tag{3.5}$$

where, with slight abuse of notation, the transformation matrices are redefined as

$$\boldsymbol{K} = \frac{2}{3}\begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \end{bmatrix} \tag{3.6}$$

and

$$\boldsymbol{K}^{-1} = \begin{bmatrix} 1 & 0 \\ -\frac{1}{2} & \frac{\sqrt{3}}{2} \\ -\frac{1}{2} & -\frac{\sqrt{3}}{2} \end{bmatrix}. \tag{3.7}$$

Note that it is implicitly assumed that the 0-component is zero when considering the reduced transformation.

Table 3.1: Definition of base values. Adopted from [4, Section 2.1.2].

Base quantity	Base value
Voltage	$V_B = \sqrt{\frac{2}{3}} V_R$
Current	$I_B = \sqrt{2} I_R$
Angular frequency	$\omega_B = \omega_1$
Apparent power	$S_B = \frac{3}{2} V_B I_B$
Impedance	$Z_B = \frac{V_B}{I_B}$
Inductance	$L_B = \frac{Z_B}{\omega_B}$
Capacitance	$C_B = \frac{1}{\omega_B Z_B}$

3.1.3 Per-Unit System

It is convenient to normalize values when considering power electronic systems (or power systems in general). The SI units are normalized with so-called *base values*. Although the base values can be arbitrarily chosen, it is common practice to use the (nominal) rated values of a system. Thus, when the system is operating at nominal conditions, most of the nominalized quantities are 1 per unit (pu).

The fundamental base values are voltage, current (or, alternatively, apparent power), and frequency. The rated voltage V_R of a grid-connected converter usually refers to the line-to-line voltage at the secondary (side) of the transformer. The base voltage is defined as the rated peak phase voltage,

$$V_B = \sqrt{\frac{2}{3}} V_R.$$

The base current is defined as the rated peak phase current (referred to the secondary as well),

$$I_B = \sqrt{2} I_R.$$

Alternatively, the rated power can instead be used as the second base value,

$$S_B = S_R.$$

The final (fundamental) base value is the frequency, which is set equal to the nominal angular fundamental frequency of the grid,

$$\omega_B = \omega_1.$$

Other base values, such as impedance, can be derived from the fundamental base values. Useful base values are summarized in Table 3.1. Time can also be normalized by multiplying it with the base (angular) frequency.

Throughout the thesis, all derivations are in SI units. However, all the implementations and results are with respect to normalized values.

3.2 Neutral-Point-Clamped Converter

The neutral-point-clamped (NPC) converter, introduced in 1981 [20], is the workhorse of the medium-voltage power electronics industry. The converter is shown in Figure 3.2. The dc-link consists of two equal bus capacitors C_d, with the dc-link voltage denoted by V_d (which is assumed to be constant). The voltages of the top and bottom capacitors are denoted by v_{top} and v_{bot}, respectively. The point between the capacitors is known as the *neutral point* and is denoted by N. Unless

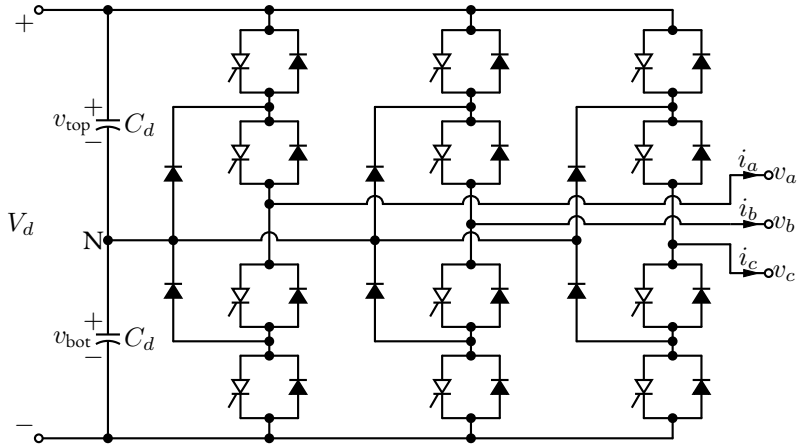


Figure 3.2: The neutral-point-clamped converter. Here, the semiconductor devices are integrated-gate-commutated thyristors with additional freewheeling diodes.

the capacitors have infinite capacitance, there will be a variation in capacitor voltages under load. The difference in capacitor voltages is referred to as the *neutral-point potential* and is defined as[1]

$$v_n = \tfrac{1}{2}(v_{\text{bot}} - v_{\text{top}}). \tag{3.8}$$

The (standard) NPC converter is a three-level topology, and can synthesize the following three voltages (with respect to N) at the output of a phase arm

$$v_p = \begin{cases} v_{\text{top}} & \text{if } u_p = 1 \\ 0 & \text{if } u_p = 0 \\ -v_{\text{bot}} & \text{if } u_p = -1, \end{cases}$$

where $p \in \{a, b, c\}$ denotes the phase and $u_p \in \{-1, 0, 1\}$ represents the switch position of a particular phase. By neglecting the variation in capacitor voltages, meaning $v_n = 0\,\text{V}$, the output at a particular phase is

$$v_p = \frac{V_d}{2} u_p. \tag{3.9}$$

The neutral-point potential is further discussed in Section 3.2.2. Table 3.2 summarizes the switching states of a phase of the NPC converter, and Figure 3.3 shows the paths for positive phase current. The negative phase current paths can be derived accordingly.

3.2.1 Voltage Vectors

The vector of the phase voltages that the converter synthesizes is

$$\boldsymbol{v}_{abc} = \frac{V_d}{2} \boldsymbol{u}_{abc}, \tag{3.10}$$

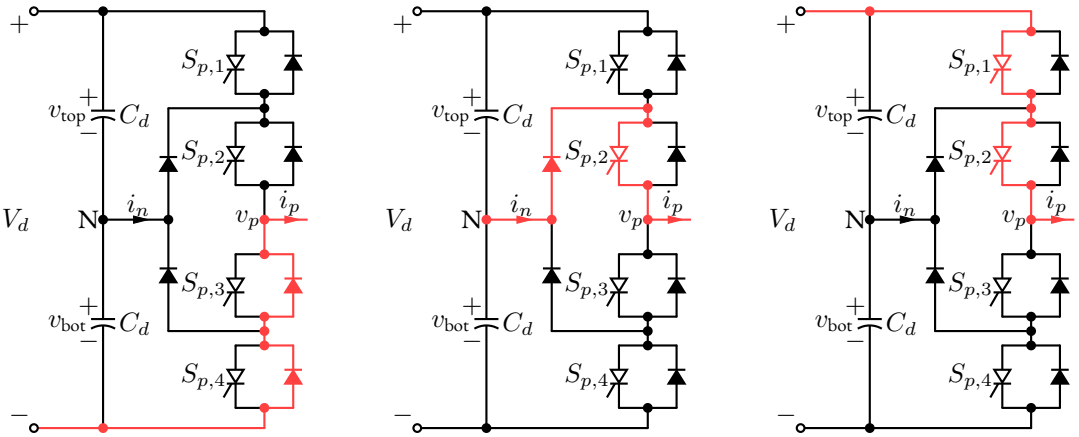
where $\boldsymbol{v}_{abc} = [v_a\ v_b\ v_c]^{\text{T}}$ and $\boldsymbol{u}_{abc} = [u_a\ u_b\ u_c]^{\text{T}} \in \{-1, 0, 1\}^3$. A three-phase three-level NPC converter can synthesize 27 different combinations of three-phase switch positions $\boldsymbol{u}_{abc}$. By applying the Clarke transformation from Section 3.1.2 to the three-phase switch positions,

$$\boldsymbol{u}_{\alpha\beta} = \boldsymbol{K}\boldsymbol{u}_{abc},$$

[1] Although the voltages are time dependent, time dependency is often dropped from variables for convenience and will be reintroduced where relevant.

Table 3.2: Switching states of an NPC converter phase arm.

Switch position	Phase voltage	Semiconductor switching state			
u_p	v_p	$S_{p,1}$	$S_{p,2}$	$S_{p,3}$	$S_{p,4}$
-1	$-\frac{V_d}{2}$	0	0	1	1
0	0	0	1	1	0
1	$\frac{V_d}{2}$	1	1	0	0



(a) $u_p = -1$ (b) $u_p = 0$ (c) $u_p = 1$

Figure 3.3: Paths for a positive phase current.

where $\boldsymbol{u}_{\alpha\beta} = [u_\alpha \; u_\beta]^{\mathrm{T}}$, it can be derived that there are only 19 vectors in the $\alpha\beta$ reference frame. These vectors are typically classified as zero, short, medium, and long vectors. It can be observed that some abc vectors result in so-called *redundant vectors* in the $\alpha\beta$ plane; three abc vectors produce the zero vector and six pairs of abc vectors constitute the short vectors. Note that the (neglected) 0-component of the short vector pairs, which represents the *common-mode voltage* v_0 at the converter output, has opposite signs. In the case that the star point of the load is floating, the potential of the star point will also be at v_0 (this can be easily derived from Figure 3.1 by using superposition). This results in the common-mode voltage not driving any phase current. In contrast, the $\alpha\beta$-components, which form the *differential-mode voltage* $\boldsymbol{v}_{\alpha\beta}$, drive phase currents.

3.2.2 Neutral-Point Potential

From Figure 3.3, it can be seen that the current flowing out of the neutral point is

$$i_n(t) = C_d \frac{\mathrm{d}v_{\text{top}}(t)}{\mathrm{d}t} - C_d \frac{\mathrm{d}v_{\text{bot}}(t)}{\mathrm{d}t}. \tag{3.11}$$

By differentiating (3.8) and inserting the result into (3.11), the evolution of the neutral-point potential is

$$\frac{\mathrm{d}v_n(t)}{\mathrm{d}t} = -\frac{1}{2C_d} i_n(t). \tag{3.12}$$

Observe from Figure 3.3 that the neutral point only conducts current when a phase is switched to it (when $u_p = 0$). The neutral-point current can thus be described by

$$i_n(t) = (1 - |u_a(t)|)i_a(t) + (1 - |u_b(t)|)i_b(t) + (1 - |u_c(t)|)i_c(t). \tag{3.13}$$

By using the fact that $i_a + i_b + i_c = 0$ holds for a floating star-connected load, the evolution of the neutral point becomes

$$\frac{\mathrm{d}v_n(t)}{\mathrm{d}t} = \frac{1}{2C_d}\left(|u_a(t)|\, i_a(t) + |u_b(t)|\, i_b(t) + |u_c(t)|\, i_c(t)\right). \tag{3.14}$$

Note that the neutral point does not conduct current when all three phases are switched to it.

During the operation of a converter system, it must be ensured that the neutral-point potential v_n is balanced and does not drift away over time; typically, some form of control over the neutral-point potential is required. Addressing the balancing of the neutral-point potential is highly challenging, since it is nonlinear [note the products in (3.14)]. Chapter 7 further discusses and addresses the neutral-point potential balancing problem. In all other chapters, the neutral-point potential is assumed to be zero (and therefore ignored).

3.3 Grid-Connected Converters

Grid-connected converters are important in many industries. A well-known application of grid-connected converters is the integration of renewable sources to the grid. In the medium-voltage drives industry, grid-connected converters are often used as an active front-end (a rectifier) in order to establish the dc-link voltage that a variable-speed drive uses as a source.

A grid-connected NPC converter is shown in Figure 3.4, where the converter is represented by switched voltage sources. The filter of a typical grid-connected converter system consists of an inductor L and (an optional) capacitor C, forming an LC filter. Unless the system is operating at a high switching frequency, the filter capacitor is typically required in order to meet the grid codes. The equivalent series resistances (ESRs) of the filter inductor and capacitor, R and R_C, are considered. The converter system is connected via a transformer to the point of common coupling (PCC). The PCC is the closest available connection a consumer has with the grid. The transformer can be represented by its leakage inductance L_t and resistance R_t. Since the grid is complex and difficult to model precisely, it is common practice to simply represent it with a grid inductance L_g, a grid resistance R_g, and a three-phase grid voltage source with a line-to-line voltage of $V_{g,\mathrm{LL}}$.

The grid is usually characterized by its short-circuit power

$$S_{\mathrm{sc}} = \frac{V_{g,\mathrm{LL}}^2}{|Z_g|},$$

where $|Z_g| = \sqrt{(\omega_1 L_g)^2 + R_g^2}$, and by its grid impedance ratio

$$k_{XR} = \frac{\omega_1 L_g}{R_g}.$$

The grid is further characterized by its short-circuit ratio, defined as

$$k_{\mathrm{sc}} = \frac{S_{\mathrm{sc}}}{S_R},$$

which is the ratio between the short-circuit power of the grid and the rated power of the converter. Ratios above 20 indicate a *strong* (or stiff) grid, whereas ratios below 10 refer to a *weak grid*. A weak grid is characterized by having a large impedance compared to that of the converter system, and causes the voltage at the PCC to vary noticeably under load. Weak grids typically have low stability margins [21].

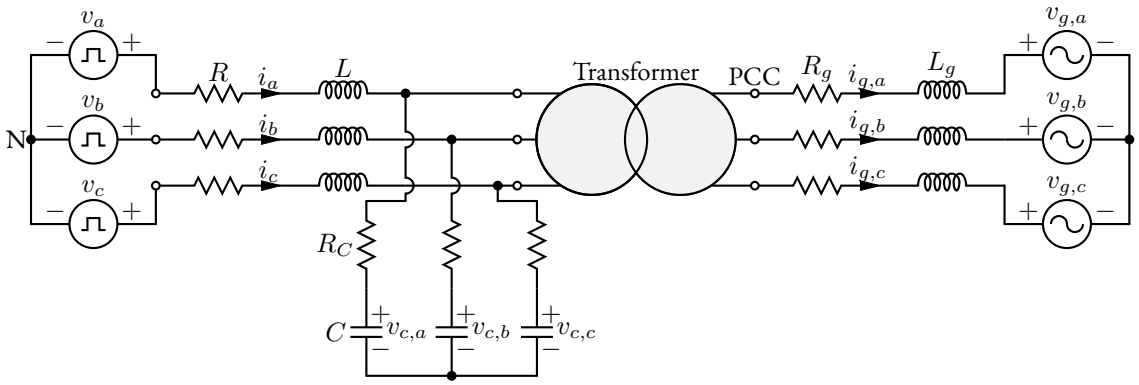


Figure 3.4: Grid-connected converter.

A grid-connected converter system is required to meet the harmonic standards imposed at the PCC. These standards are required so that systems in a grid network do not harm each other. Although these standards are not universal, a popular and widely adopted standard is the IEEE Std 519 [22, 23], which establishes current and voltage harmonic limits (also called distortion limits). According to [22, Table 10.3], which details the current limits for systems up to 69 kV, stricter harmonic limits are required for weaker grids. The harmonic limit also becomes stricter for harmonics of a higher order. The limits on even-order harmonics are particularly stringent, and no dc offset is allowed. The voltage harmonic standard [22, Table 11.1], although not as detailed as the current harmonic standard, requires that all individual voltage harmonics must be below a certain limit. There is also a limit on the total voltage distortion. Another popular standard is the IEC 61000, which focuses more on the voltage distortion limits.

3.3.1 Modelling of a Grid-Connected Converter

For the grid-connected converter in Figure 3.4, the state variables are the converter currents $\boldsymbol{i}_{abc} = [i_a\ i_b\ i_c]^{\mathrm{T}}$, grid currents $\boldsymbol{i}_{g,abc} = [i_{g,a}\ i_{g,b}\ i_{g,c}]^{\mathrm{T}}$, and capacitor voltages $\boldsymbol{v}_{c,abc} = [v_{c,a}\ v_{c,b}\ v_{c,c}]^{\mathrm{T}}$. It is assumed that the dc-link voltage is constant at its nominal value, and the neutral-point potential has no fluctuations. Recall from (3.10) that the switched voltage source $\boldsymbol{v}_{abc} = [v_a\ v_b\ v_c]^{\mathrm{T}}$ (the three-phase output voltage of the NPC converter) is simply the switch positions $\boldsymbol{u}_{abc} = [u_a\ u_b\ u_c]^{\mathrm{T}}$ scaled by $\frac{V_d}{2}$. Note the input $\boldsymbol{u}_{abc}$ is restricted to the set $\{-1, 0, 1\}^3$. The grid voltage source $\boldsymbol{v}_{g,abc}$ is treated as a (known) disturbance.

By using the Clarke transformation $\boldsymbol{\xi}_{\alpha\beta} = \boldsymbol{K}\boldsymbol{\xi}_{abc}$ to map the three-phase abc variables to the $\alpha\beta$ variables and analyzing the circuit of Figure 3.4, the following system of differential equations is obtained

$$\frac{\mathrm{d}\boldsymbol{i}(t)}{\mathrm{d}t} = -\frac{R + R_C}{L}\boldsymbol{i}(t) + \frac{R_C}{L}\boldsymbol{i}_g(t) - \frac{1}{L}\boldsymbol{v}_c(t) + \frac{V_d}{2L}\boldsymbol{K}\boldsymbol{u}_{abc}(t) \tag{3.15a}$$

$$\frac{\mathrm{d}\boldsymbol{i}_g(t)}{\mathrm{d}t} = \frac{R_C}{L_{gt}}\boldsymbol{i}(t) - \frac{R_{gt} + R_C}{L_{gt}}\boldsymbol{i}_g(t) + \frac{1}{L_{gt}}\boldsymbol{v}_c(t) - \frac{1}{L_{gt}}\boldsymbol{v}_g(t) \tag{3.15b}$$

$$\frac{\mathrm{d}\boldsymbol{v}_c(t)}{\mathrm{d}t} = \frac{1}{C}\boldsymbol{i}(t) - \frac{1}{C}\boldsymbol{i}_g(t), \tag{3.15c}$$

where $\boldsymbol{i} = [i_\alpha\ i_\beta]^{\mathrm{T}}$, $\boldsymbol{i}_g = [i_{g,\alpha}\ i_{g,\beta}]^{\mathrm{T}}$, $\boldsymbol{v}_c = [v_{c,\alpha}\ v_{c,\beta}]^{\mathrm{T}}$, and $\boldsymbol{v}_g = [v_{g,\alpha}\ v_{g,\beta}]^{\mathrm{T}}$. The resistances and inductances of the grid and transformer are lumped together as $R_{gt} = R_g + R_t$ and $L_{gt} = L_g + L_t$, respectively. For the derivation of (3.15), refer to Appendix B. Note that, except for the three-phase switch position $\boldsymbol{u}_{abc}$, all variables are referred to the $\alpha\beta$ coordinate system. For convenience, the

Table 3.3: Rated values of the medium-voltage converter system.

Parameter	Symbol	Value
Rated line-to-line voltage (at the secondary)	V_R	3150 V (rms)
Rated phase current (at the secondary)	I_R	1649.6 A (rms)
Rated angular frequency	ω_R	$\omega_1 = 2\pi 50\,\mathrm{rad/s}$

Table 3.4: System parameters of the medium-voltage case study.

Parameter	Symbol	Value	Per unit
Converter			
Dc-link voltage	V_d	4840 V	1.8818 pu
Rated apparent power	S_R	9 MVA	1 pu
Half dc-link capacitance	C_d	9.9 mF	3.4290 pu
Filter inductance	L	350 µH	0.0997 pu
Filter capacitance	C	420 µF	0.1455 pu
Filter inductor ESR	R	0.3 mΩ	0.000 27 pu
Filter capacitor ESR	R_C	4 mΩ	0.0036 pu
Transformer inductance	L_t	526.41 µH	0.15 pu
Transformer resistance	R_t	16.54 mΩ	0.015 pu
Grid			
Grid-side inductance	L_g	349.19 µH	0.0995 pu
Grid-side inductor ESR	R_g	10.97 mΩ	0.010 pu
Fundamental frequency	f_1	50 Hz	1 pu
Grid voltage (line-to-line)	$V_{g,\mathrm{LL}}$	3150 V (rms)	1.2247 pu
Rated phase current	I_R	1649.6 A (rms)	0.7071 pu

$\alpha\beta$ subscript is dropped from the respective variables, whereas abc variables will have their subscript explicitly stated.

3.3.2 Medium-Voltage Case Study

A 9 MVA grid-connected NPC converter with an LC filter is (primarily) used as a case study throughout this thesis. The rated values are shown in Table 3.3, whereas the system parameters are shown in Table 3.4. Both the short-circuit ratio k_{sc} and grid impedance ratio k_{XR} are assumed to be 10. The base values are set equal to their peak phase rated values in accordance with the per-unit system in Section 3.1.3. The system possesses two resonant frequencies at 262 Hz and 491 Hz.

3.4 Optimized Pulse Patterns

Optimized pulse patterns (OPPs) are a pulse-width modulation technique (hence they are also known as synchronous optimal pulse-width modulation) that is primarily used at low switching frequencies. One of the earliest uses of OPPs was in 1977 [24], where it was proposed for two-level converters. OPPs offer a number of benefits over conventional pulse-width modulation (PWM) methods such as well-known carrier-based PWM (CB-PWM). Specifically, OPPs have superior harmonic performance and offer remarkable flexibility. OPPs are optimal in the sense that switching

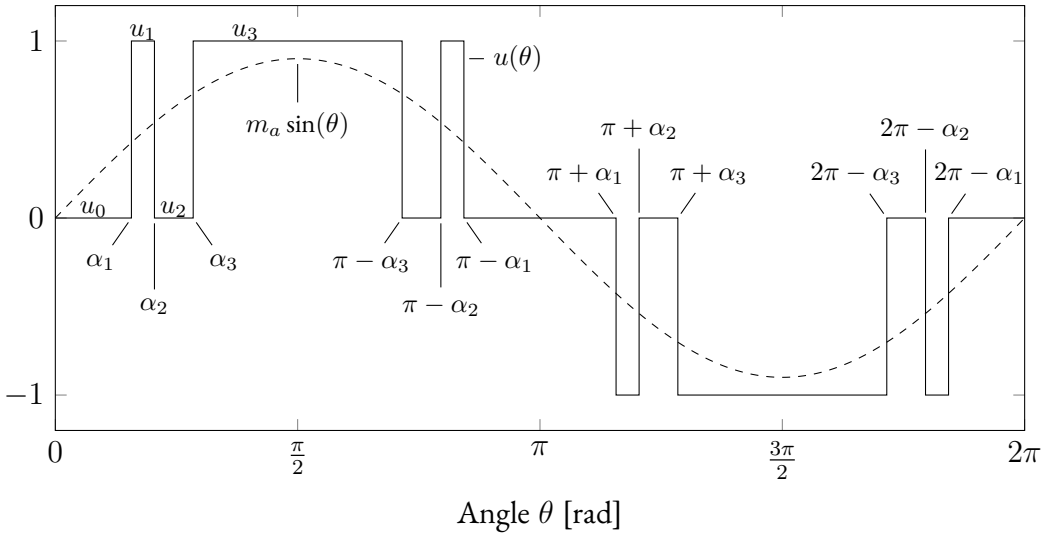


Figure 3.5: Single-phase pulse pattern $u(\theta)$.

angles are computed (offline) that minimize an objective function which, typically, relates to harmonic distortion. Since the OPP problem formulation involves an optimization problem, a large degree of freedom is permitted regarding the characteristics of a pulse pattern.

Only three-level pulse patterns are considered. For more further reading on pulse patterns with more levels, see [4, Section 3.4.3] and [25].

3.4.1 Pulse Pattern

Consider the single-phase pulse pattern $u(\theta)$, where $\theta = \omega_1 t$, as shown in Figure 3.5. Similar to other PWM methods, the fundamental component of the OPP is $m_a \sin(\theta)$, which is the modulating signal that the pulse pattern approximates, where m_a is the modulation index. The pulse pattern is periodic with a period of 2π with quarter-wave symmetry (typically) imposed. Denote with d the so-called *pulse number*, which represents the number of switching transitions within a quarter wave. The pulse number relates the fundamental frequency to the device switching frequency with $f_{\text{sw}} = df_1$ (the switching frequency is synchronized with the fundamental frequency). The pulse pattern has d (primary) switching angles (that occur within the first quarter wave) denoted by α_i, for $i = 1, 2, \ldots, d$; due to the quarter-wave symmetry, the entire pulse pattern can be characterized by these switching angles. At each switching instant, the so-called *switch position* that the pulse pattern assumes is denoted by $u_i \in \{-1, 0, 1\}$. From this, the change in switch position (called a *switching transition*) is defined as $\Delta u_i = u_i - u_{i-1} \in \{-1, 1\}$. It can be observed that for $\theta \in [0, \frac{\pi}{2}]$, a pulse pattern can be described as

$$u(\theta) = u_0 + \sum_{i=1}^{d} \Delta u_i h(\theta - \alpha_i), \tag{3.16}$$

where $h(\theta)$ is the well-known (unit) step function (see Appendix C for a brief review on the step function). Note that due to quarter-wave symmetry, the initial switch position u_0 is required to be zero. Assume that switching is performed between 0 and 1 within the first quarter wave, resulting in a switching transition being described as $\Delta u_i = (-1)^{i+1}$.

The next section analyzes the harmonic content of a pulse pattern in order to formulate the harmonic distortion that a pulse pattern results in.

3.4.2 Harmonic Analysis

Fourier Series of a Pulse Pattern

Since a pulse pattern is periodic, it can be represented by a Fourier series,

$$u(\theta) = \frac{a_0}{2} + \sum_{n=1}^{\infty} \left(a_n \cos(n\theta) + b_n \sin(n\theta) \right),$$

with the Fourier coefficients

$$a_n = \frac{1}{\pi} \int_0^{2\pi} u(\theta) \cos(n\theta) \mathrm{d}\theta, \text{ for } n \geq 0$$
$$b_n = \frac{1}{\pi} \int_0^{2\pi} u(\theta) \sin(n\theta) \mathrm{d}\theta, \text{ for } n \geq 1.$$

Since a pulse pattern is an odd function with quarter-wave symmetry, it follows that $a_n = 0$ for all n and $b_n = 0$ for all even n [26, Section 16.3]. This leads to the Fourier series representation reducing to

$$u(\theta) = \sum_{n=1,3,5,\ldots}^{\infty} \hat{u}_n \sin(n\theta) \tag{3.17}$$

where the amplitude of the nth harmonic is

$$\hat{u}_n = \frac{4}{\pi} \int_0^{\frac{\pi}{2}} u(\theta) \sin(n\theta) \mathrm{d}\theta, \text{ for any odd } n. \tag{3.18}$$

Note that a pulse pattern with half-wave symmetry has no even-order harmonics. Inserting (3.16) into (3.18) results in

$$\hat{u}_n = \frac{4}{\pi} \int_0^{\frac{\pi}{2}} \sum_{i=1}^{d} \Delta u_i h(\theta - \alpha_i) \sin(n\theta) \mathrm{d}\theta,$$

and by knowing that the step function changes the lower bound of the integral,

$$\hat{u}_n = \frac{4}{\pi} \sum_{i=1}^{d} \Delta u_i \int_{\alpha_i}^{\frac{\pi}{2}} \sin(n\theta) \mathrm{d}\theta,$$

it follows that the nth harmonic is

$$\hat{u}_n = \frac{4}{n\pi} \sum_{i=1}^{d} \Delta u_i \cos(n\alpha_i), \tag{3.19}$$

where the fact that $\cos(n\frac{\pi}{2}) = 0$ for any odd n has been used. Observe from (3.19) that the largest possible fundamental component (the highest modulation index) is $\frac{4}{\pi}$, which is achieved during square-wave switching ($d = 1$ and $\alpha_1 = 0$ rad).

Thus far, the characteristics of a three-phase system has not been taken into account when considering the harmonics. A three-phase pulse pattern is symmetrical in the three phases with a positive phase sequence,

$$\boldsymbol{u}_{abc}(\theta) = \begin{bmatrix} u(\theta) \\ u(\theta - \frac{2\pi}{3}) \\ u(\theta + \frac{2\pi}{3}) \end{bmatrix},$$

where

$$u(\theta \pm \frac{2\pi}{3}) = \sum_{n=1,3,5,\ldots}^{\infty} \hat{u}_n \sin(n\theta \pm n\frac{2\pi}{3}).$$

It can be seen that the third-order harmonics (with n being a multiple of three) of the three phases are in phase. The third-order harmonics constitute common-mode voltages, which do not induce phase currents. Therefore, third-order harmonics do not contribute to the harmonic distortion of phase current.

Harmonic Distortion

The harmonic distortion that an OPP will result in needs to be quantified. Usually, the harmonic distortion of the OPP itself is not of interest; instead the so-called *weighted* harmonic distortion is required.

Consider the transfer function

$$\frac{\xi(s)}{u(s)} = H(s)$$

of a general load, where s is the complex frequency, u is the pulse pattern, and ξ is some converter quantity (for example, an inductor current or capacitor voltage). The amplitude of nth harmonic of ξ is the harmonic $\hat{u}_n$ of the pulse pattern that is weighted with $|H(nj\omega_1)|$:

$$\hat{\xi}_n = |H(nj\omega_1)|\,\hat{u}_n,$$

where j is the imaginary unit. A useful and widely used metric (in power systems) for the distortion of a quantity is the *total demand distortion* (TDD), which is defined as

$$\xi_{\text{TDD}} = \frac{1}{\sqrt{2}\xi_{\text{nom}}}\sqrt{\sum_{i\neq 1}\hat{\xi}_n^2}, \tag{3.20}$$

where ξ_{nom} is the nominal rms value of ξ. Alternatively, the total harmonic distortion can also be used as a metric, where $\sqrt{2}\xi_{\text{nom}}$ is replaced by the amplitude of the fundamental component $\hat{\xi}_1$.

As an example, consider a voltage source in series with an inductive load (where the resistance is assumed to be negligible) that is connected to the output of a converter. Systems that can be modelled as such are grid-connected converters (without a filter capacitor) and electrical machines. Recall that the phase output voltage of the converter is $v = \frac{V_d}{2}u$. The transfer function (for $s \neq j\omega_1$) for the pulse pattern to the inductor current is

$$H(s) = \frac{1}{sL}\frac{V_d}{2},$$

leading to the amplitude of the nth harmonic of the inductor current being

$$\hat{i}_n = \frac{1}{\omega_1 L}\frac{V_d}{2}\frac{\hat{u}_n}{n}. \tag{3.21}$$

From the definition of the TDD in (3.20), and with the $\hat{u}_n$ given by (3.19) and $\hat{i}_n$ given by (3.21), the TDD of the inductor current is

$$I_{\text{TDD}} = \frac{\sqrt{2}V_d}{\pi\omega_1 L}\frac{1}{I_{\text{nom}}}\sqrt{\sum_{n=5,7,11,\ldots}\left(\frac{1}{n^2}\sum_{i=1}^{d}\Delta u_i\cos(n\alpha_i)\right)^2}, \tag{3.22}$$

where I_{nom} is the nominal inductor current. Note that only odd non-third-order harmonics are considered, since all other harmonics are either zero or do not contribute to the TDD. Observe that the squared TDD is proportional to[2]

$$I_{\text{TDD}}^2 \propto \sum_{n=5,7,11,\ldots} \left(\frac{1}{n^2} \sum_{i=1}^{d} \Delta u_i \cos(n\alpha_i) \right)^2,$$

where it is seen that the harmonics of the pulse pattern of a higher order are weighted less, since they will be attenuated by the inductor. In the case of a grid-connected converter with an LC filter, the TDD of the grid current will be of concern, and the harmonics will be weighed differently to those of an inductive load.

With the expression of the current TDD as a function of the switching angles α_i, the pulse pattern can be optimized to have minimum harmonic distortion of the current.

3.4.3 Optimization Problem

The minimization of the harmonic distortion leads to a nonlinear optimization problem [as shown in (2.1)]. The objective function is proportional to the squared TDD of the current,

$$J(\boldsymbol{\alpha}) = \sum_{n=5,7,11,\ldots} \left(\frac{1}{n^2} \sum_{i=1}^{d} \Delta u_i \cos(n\alpha_i) \right)^2, \tag{3.23}$$

with the vector of switching angles $\boldsymbol{\alpha} = [\alpha_1 \ \alpha_2 \ \cdots \ \alpha_d]^{\mathrm{T}}$ as the decision variable. During the operation of a converter system, the fundamental component is required to be equal to the desired modulation index, resulting in an equality constraint. From (3.19), with $\hat{u}_1 = m_a$, the constraint is

$$\frac{4}{\pi} \sum_{i=1}^{d} \Delta u_i \cos(n\alpha_i) = m_a. \tag{3.24}$$

In order to guarantee that the switching transitions result in a feasible pulse pattern, it is required that the switching transitions are nonnegative, in ascending order, and occur before the quarter-wave point. This results in the following $d + 1$ inequality constraints:

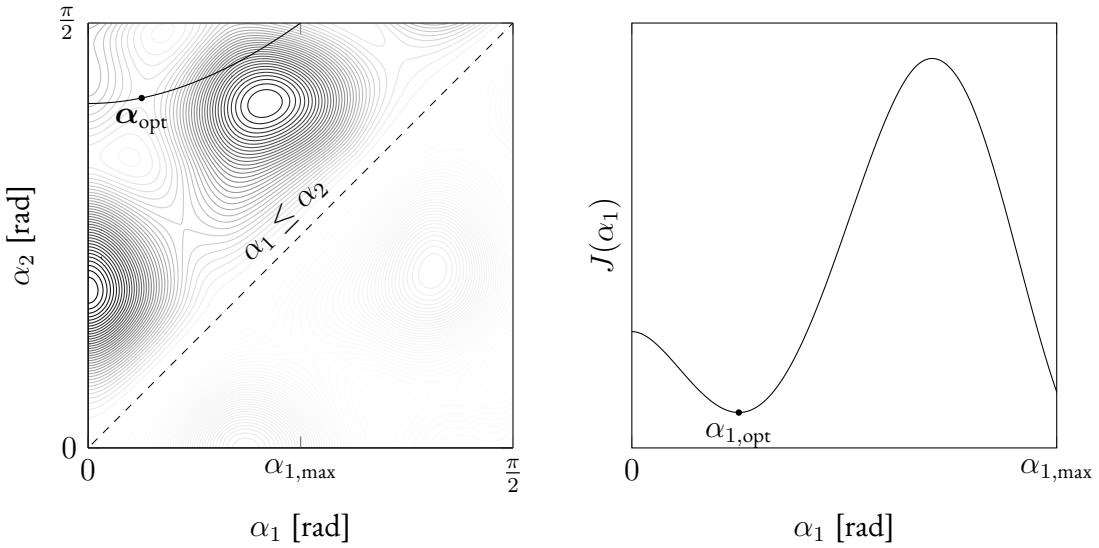
$$0 \leq \alpha_1 \leq \alpha_2 \leq \ldots \leq \alpha_d \leq \frac{\pi}{2}. \tag{3.25}$$

The resulting nonlinear program is

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \quad & J(\boldsymbol{\alpha}) && \text{(3.26a)} \\ \text{subject to} \quad & (3.24) && \text{(3.26b)} \\ & (3.25), && \text{(3.26c)} \end{aligned}$$

which is illustrated in Figure 3.6. It is easy to see that the objective function is nonconvex, which can be expected due to the trigonometric terms. Solving (3.26) to global optimality requires significant effort, especially for high pulse numbers. The optimization problem is usually repeated with different initial conditions to ensure global optimality.

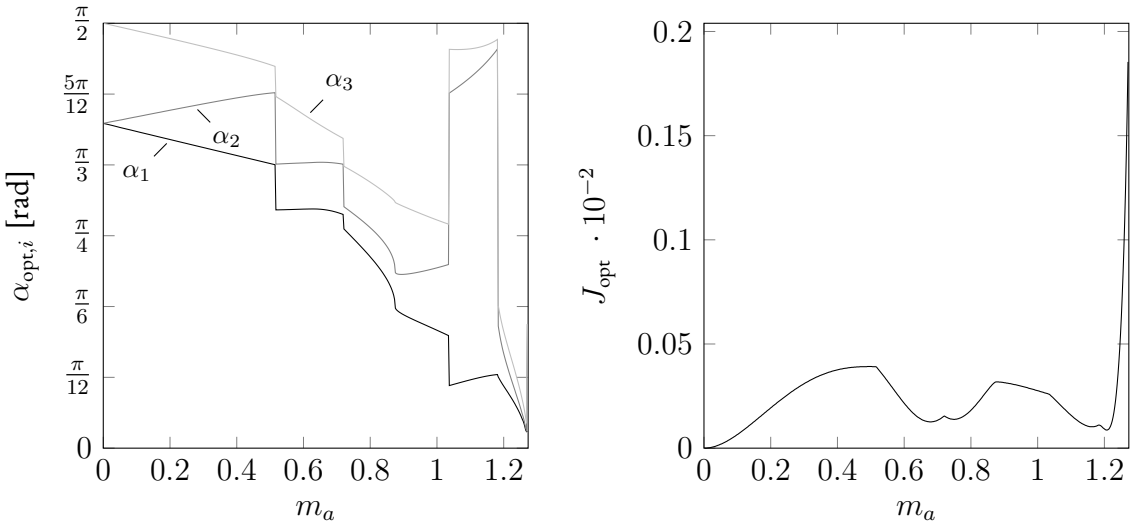
[2] The same relation results if the total harmonic distortion is used as a metric.



(a) The contours of the objective function. The equality constraint refers to $m_a = 0.9$. Only the region that satisfies the inequality constraints is shown.

(b) Cost of the objective function when moving along the equality constraint as a function of α_1.

Figure 3.6: Visualization of the optimization problem underlying OPPs with pulse number $d = 2$.



(a) The switching angles as a function of the modulation index.

(b) The optimal cost as a function of the modulation index.

Figure 3.7: The optimal switching angles $\boldsymbol{\alpha}_{\text{opt}}$ and optimal cost J_{opt} for pulse number $d = 2$.

Due to the high computational burden that is required when computing OPPs, the optimization is done offline. For a given pulse number, the OPPs are computed over a range of modulation indices $m_a \in [0, \frac{4}{\pi}]$, with the (optimized) switching angles then being stored in a lookup table. During real-time operation of a converter system, the OPP corresponding to the desired modulation index is read-out. As an example, consider a pulse pattern with pulse number $d = 3$ to be optimized. The `MultiStart` method from Matlab is used with 100 initial conditions that are generated by a Halton sequence. The sequential quadratic programming (`sqp` in Matlab) algorithm is selected. Harmonics up to the 200th are considered, as the impact of higher order harmonics on the TDD

Table 3.5: System parameters of a typical medium-voltage system.

Parameter	Symbol	Value
Dc-link voltage	V_d	1.9 pu
Inductance	L	0.25 pu
Resistance	R	0.025 pu
Fundamental frequency	f_1	50 Hz
Load voltage source (line-to-line)	V_{LL}	1.2247 pu
Rated phase current	I_R	0.7071 pu

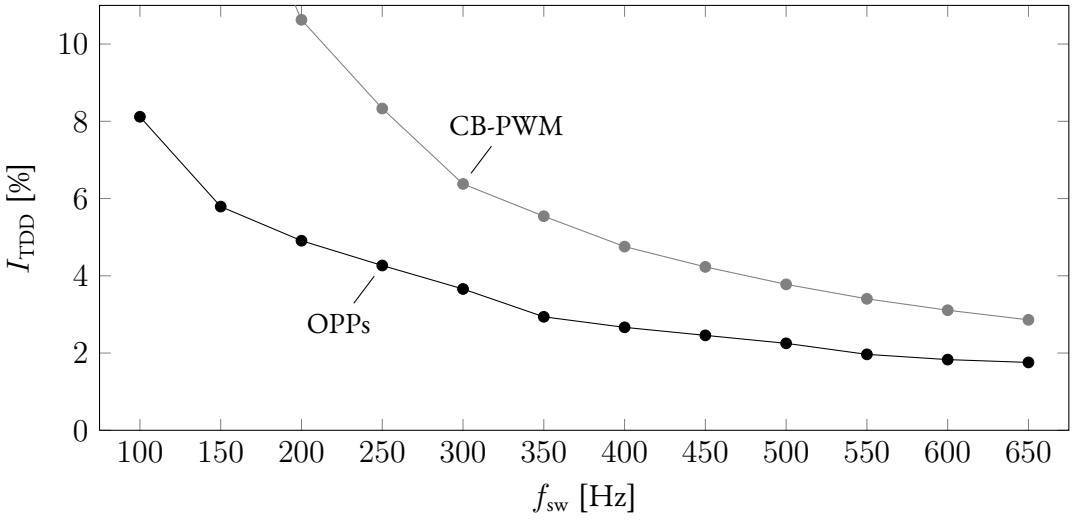


Figure 3.8: The TDD of the current for OPPs and CB-PWM as a function of the switching frequency at a modulation index of $m_a = 1.111$.

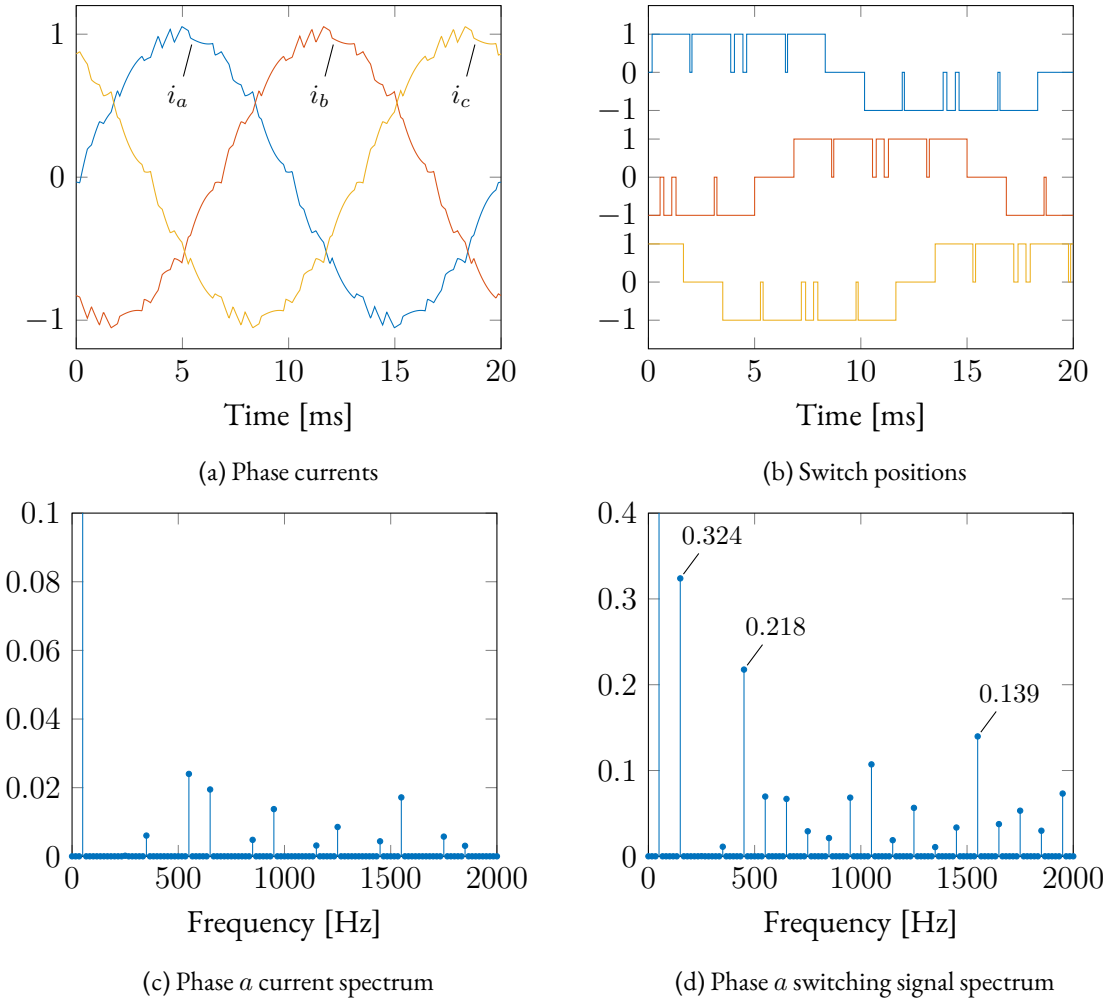
rapidly decreases. The optimal switching angles $\boldsymbol{\alpha}_{\text{opt}}$ and cost J_{opt} resulting from the optimization are shown in Figure 3.7. For noncommercial alternatives to solve the optimization problem, Python and Julia offer a number of libraries that can be called, such as IPOPT [27].

The OPP problem formulation that has been used thus far can be seen as a minimum representation of the OPP problem; pulse patterns are highly flexible, and additional terms can be added to the objective function or to the constraints. For example, constraints may be placed on certain harmonics (say, the 5th and 7th) so that they do not exceed a certain amplitude. Additional freedom can be obtained when relaxing the quarter-wave symmetry. This can result in lower harmonic distortion, which has recently been investigated in [28].

3.4.4 Comparison with Carried-Based Pulse-Width Modulation

In order to motivate the use of OPPs, this section demonstrates the harmonic performance of OPPs against that of CB-PWM. The CB-PWM technique that is considered is asymmetric-regularly-sampled PWM with space-vector centering, which is a standard method in power electronics. For brevity, these details are omitted and the interested reader is referred to [29] for further information on PWM, and [4, Section 3.3] for an accessible overview.

As a case study, typical per-unit values of a medium-voltage first-order converter system are used, which can represent an electrical machine or a grid-connected system. The higher-order case study from Section 3.3.2 is not used due to the severe limitation the resonance will impose on lowering the switching frequency of CB-PWM. The parameters are summarized in Table 3.5, where the



(a) Phase currents

(b) Switch positions

(c) Phase a current spectrum

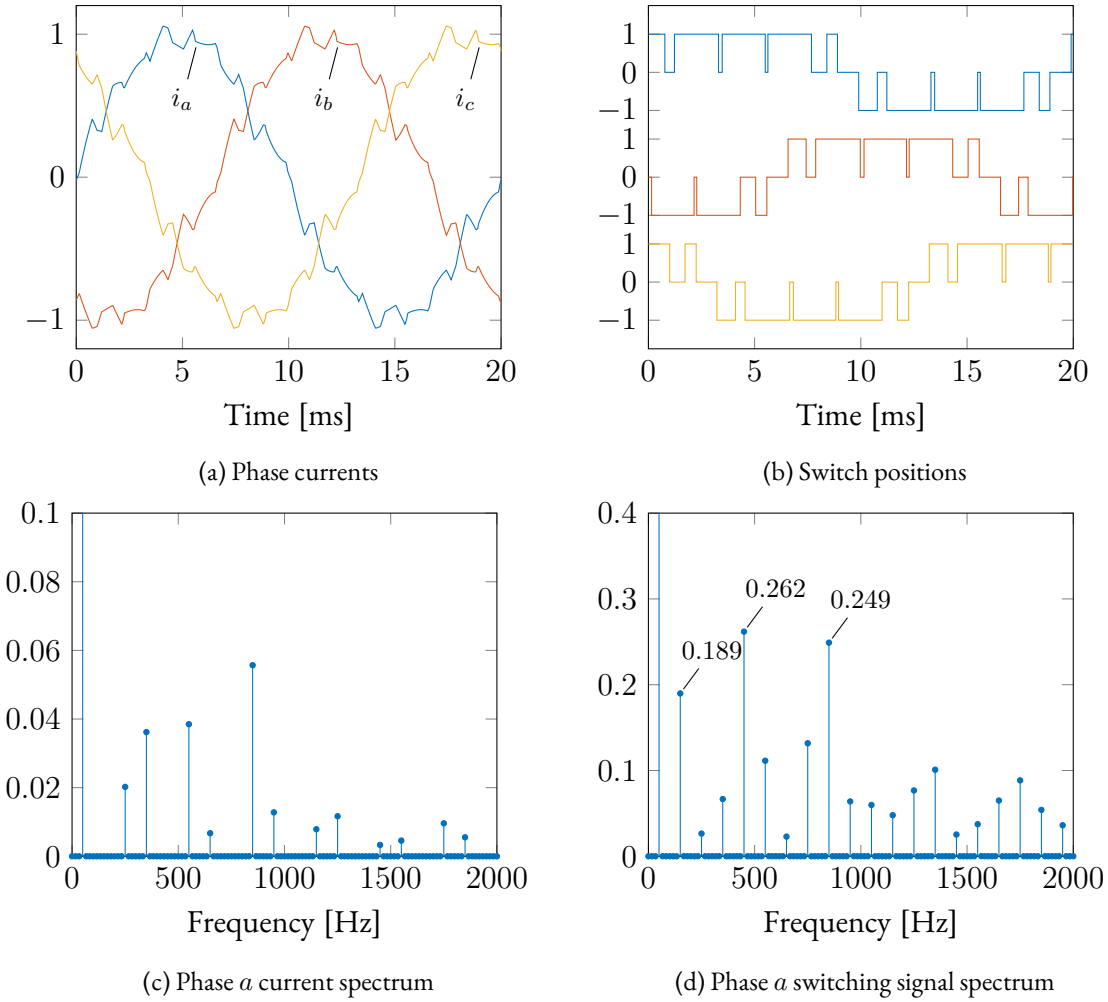
(d) Phase a switching signal spectrum

Figure 3.9: The waveforms (in pu) of an OPP with pulse number $d = 5$. The TDD of the current is $I_{\text{TDD}} = 4.27\,\%$.

load voltage source may represent a grid voltage or the counter electromotive force of an electrical machine.

The operating conditions for all tests are at rated current and unity power factor, resulting in a modulation index of $m_a = 1.111$.[3] In Figure 3.8, the TDD of the current versus the device switching frequency is shown for OPPs and CB-PWM. Switching frequencies f_{sw} between 200 Hz and 350 Hz are of particular relevance for medium-voltage applications, as this is typically the operating range for these systems (assuming a fundamental frequency of 50 Hz). It can be observed that OPPs significantly outperform CB-PWM in terms of harmonic performance at very low switching frequencies; reductions of 54 % and 49 % at switching frequencies of 200 Hz and 250 Hz, respectively, are achieved relative to CB-PWM. A consequence of the lower harmonic distortion that OPPs possess is that the converter can operate at a lower switching frequency while still having acceptable TDD levels. In turn, this leads to lower switching losses in the power converter. Thus, from another viewpoint, OPPs lower the losses in the system at a given distortion level when compared to CB-PWM. It is worth mentioning that CB-PWM is not suitable for very low switching frequencies, with its performance deteriorating. Although not visible in Figure 3.8, CB-PWM yielded better harmonic performance at a switching frequency of $f_{\text{sw}} = 100$ Hz ($I_{\text{TDD}} = 12.43\,\%$) than at

[3] For regularly-sampled PWM, the fundamental component of the pulse-width modulated switching pattern will be *slightly* different to that of the modulation index due to the sample-and-hold operation.



(a) Phase currents

(b) Switch positions

(c) Phase a current spectrum

(d) Phase a switching signal spectrum

Figure 3.10: The waveforms (in pu) of CB-PWM with a carrier frequency of 450 Hz. The TDD of the current is $I_{\text{TDD}} = 8.33\,\%$.

$f_{\text{sw}} = 150\,\text{Hz}$ ($I_{\text{TDD}} = 15.43\,\%$). This is due to the phase of the carrier, which is kept at zero for all simulations, having a significant influence on the TDD of the current at low switching frequencies [30].

For more insight into the mechanisms that OPPs exploit to achieve such low harmonic distortions, consider the steady-state waveforms for an OPP and CB-PWM in Figures 3.9 and 3.10, respectively. The device switching frequency is $f_{\text{sw}} = 250\,\text{Hz}$ in both cases. From the spectrum of the phase a switching signal u_a for the OPP and CB-PWM (Figures 3.9d and 3.10d, respectively), it is seen that the OPP shifts significant amount of harmonic power into the 3rd harmonic (0.324) when compared to CB-PWM (0.189). At the 9th harmonic, the OPP has slightly less harmonic power (0.218) than CB-PWM (0.262). The power in all third-order harmonics (which establish the common-mode components) of the OPP is 0.4353, which is noticeably higher than the 0.4033 of CB-PWM.[4] Recall that third-order harmonics do not contribute to the TDD of the phase current. Regarding non-third-order harmonics, it can been observed that the OPP tends to shift harmonic power into higher-order harmonics. Recall that higher-order harmonics are weighted less (in other words, they contribute less to the current TDD), since they are attenuated by the inductor. The largest (non-third-order) harmonic of the OPP is the 31st harmonic (0.139), whereas that of CB-PWM is the 19th harmonic (0.249). The 19th harmonic of CB-PWM is easily noticeable in the

[4] The power in the third-order harmonics is calculated as $\sqrt{\sum_{n=3,9,15,\ldots} \hat{u}_n^2}$.

current spectrum in Figure 3.10c.

However, OPPs entail certain challenges; designing a suitable controller for a system that is modulated with OPPs is a difficult (and somewhat unexplored) topic. Chapter 4 discusses related control techniques, where some of the underlying difficulties of OPPs are highlighted. In Chapter 5, a new OPP-based control technique is proposed that overcomes the limitations of existing methods.

Chapter 4

Model Predictive Control and Existing Control Schemes

This chapter first provides an introduction to model predictive control; specifically, its origin and control principle are discussed. This is followed by a discussion on finite-control-set model predictive control, a very popular technique in the power electronics research community. OPP-based control methods are then introduced. Special attention is given to model predictive pulse pattern control, which is a state-of-the-art control technique used in the medium-voltage power electronics industry. Its primary limitation is highlighted, which the control technique proposed in Chapter 5 addresses. The chapter concludes with a review of some control techniques that also attempt to address the limitation of model predictive pulse pattern control. However, these methods do have some underlying drawbacks, which are discussed, that either make them impractical or limit their performance.

Chapter Contents

4.1 Introduction to Model Predictive Control

One of the largest contributions to optimal control was by Kalman in 1960 [31], where the notions of controllability and observability were introduced along with a rigorous and complete analysis of the regulator problem. This gave rise to the well-known *linear-quadratic regulator* (LQR), which is widely used to control linear multiple-input multiple-output (MIMO) systems. The LQR problem involves calculating the optimal behaviour of a system over an *infinite* horizon in the time domain. Solving the control problem yields a (closed-loop) state-feedback matrix.

However, the (classic) LQR is limited to linear systems without any input or state constraints. The limitations of the LQR motivated the research and development of *model predictive control* (MPC) techniques which can address the control of nonlinear systems while enforcing constraints on states and inputs. The first variants of MPC appeared in the late 1970s [32]. Since then, MPC has become a mature control technique and is widely used in the process industry, the food processing industry, and even with some applications in the aerospace and defence industry [33, Table 6].

Although some early research of MPC for power electronics was conducted in the 1980s [34], only since the mid 2000s has it been readily adopted in the research community of power electronics, where it has become a popular and active research topic.[1] This late adoption of MPC in power electronics can be attributed to the fast dynamics of power electronic systems, which typically require short sampling intervals (within the microsecond range). In the past, these short sampling intervals made MPC an infeasible option due to the limited computational power that was available. With the recent advancements in the processing power of embedded systems, MPC has become a viable and attractive control technique for power electronic applications. Since MPC is a MIMO controller, it is well suited for power electronic systems (which are typically MIMO systems). Furthermore, being able to impose constraints on the inputs enables the controller to take the limitations of the system into account; no anti-windup schemes are required. Further constraints can be placed on system states to ensure that the converter system is operating within safe limits. Finally, nonlinearities can be addressed by MPC in a systematic manner, as the nonlinear behaviour can be incorporated in the internal dynamic model. However, it should be stressed that nonlinearities do increase the complexity of the problem. For this reason there is often a distinction between linear and nonlinear MPC techniques.

Traditionally, and nearly exclusively, the control problem of MPC techniques is formulated in the discrete-time domain. Specifically, the internal dynamic model of the controller is discrete and the control problem is formulated accordingly. However, the MPC technique proposed in Chapter 5 is formulated in the continuous-time domain (but operates at discrete instants). Although the proposed control technique follows all the principles of MPC (which is explained soon), the method does differ moderately from discrete-time MPC. In order to avoid confusion and keep the scope limited, details regarding traditional (discrete-time) MPC are omitted. Readers interested in the formulation of traditional discrete-time MPC are referred to the classic textbooks [36, 37]. The continuous-time MPC formulation is further discussed in Chapter 5 within the context of the proposed control method. For now, only a brief overview of the principles of MPC is given; Section 5.5 revisits these concepts with appropriate illustrations.

All model predictive controllers consist of an *internal dynamic model* to *predict* the future states of a system over a *finite* prediction horizon as a function of the system inputs.[2] A constrained optimal control problem is formulated, with the control objectives mapped into an *objective function*, and with *constraints* representing (physical) limitations as well as operating limits. Solving the underlying *optimization problem* yields an optimal (open-loop) sequence of manipulated variables over

[1] The interested reader is referred to [35] for a recent survey paper on MPC for power electronics.

[2] MPC techniques are also available with infinite horizons.

the prediction horizon. In order to provide feedback and make the system robust to disturbances and modelling errors, only the (optimal) manipulated variable during the current sampling interval is applied. At each subsequent sampling instant, the prediction horizon is shifted, new measurements are taken, and the sequence of manipulated variables is re-optimized; this is known as the *receding horizon* policy [37, Section 1.1]. The receding horizon is synonymous with MPC, hence it is also known as receding horizon control.

4.2 Finite-Control-Set Model Predictive Control

An MPC technique that has gained significant traction in the power electronics research community is finite-control-set (FCS) MPC [38], where it is arguably the most popular MPC technique due to its intuitive nature. Typically, a power electronic system includes a pulse-width modulator that converts a real-valued signal (usually the modulating signal that is determined by the controller) into a discrete-valued pulse-width modulated signal that is used to switch semiconductor devices. However, a pulse-width modulator is a nonlinear device, and its switched behaviour is nontrival to model efficiently. Most control schemes usually assume an averaged modulator, reducing it to a constant gain, which can easily be included in the internal model of the controller. The averaged model ignores the harmonics of PWM and, therefore, the ripple of the converter quantities. This approach is suitable if the switching frequency of the converter is high enough. At low switching frequencies, however, the ripple becomes significant, and the switching nature of the converter should be considered. So-called *direct* MPC methods overcome this problem by combining the modulation stage with the controller; a pulse-width modulator is not required.

FCS-MPC is a discrete-time MPC method and predicts the system states as a function of the input over a prediction horizon of N_p discrete-time steps. Note that due to the controller handling the modulation stage, the manipulated variable is restricted to a finite set of integers $\boldsymbol{\mathcal{U}} \subset \mathbb{Z}^{n_u}$, where n_u is the number of inputs. This implies that the optimization problem is an integer program. Although integer programs seem conceptually easy to solve, since the (finite) feasible set can be enumerated in order to find the optimal solution, integer programs are actually significantly harder to solve than a problem with continuous variables. Note that the number of elements in the nonconvex feasible set increases exponentially with an increasing number of prediction steps; there is a combinatorial explosion of possible solutions over the prediction horizon, making an enumeration-based approach intractable.[3] In fact, integer programs are known to be nondeterministic polynomial-time hard [39], meaning that it is highly unlikely that an algorithm will ever exist that can solve the problem efficiently for all problem classes; the computational time required to find the optimal solution will eventually grow exponentially as the problem size increases.

Nonetheless, for the problem sizes considered in power electronics, the underlying integer program of FCS-MPC can be solved in real-time. Typically, most uses of FCS-MPC only considers a single-step prediction, which is trivial to solve (it is feasible to enumerate all possible solutions). On the other hand, long prediction horizons (that is, multiple prediction steps) offer a considerable improvement in harmonic performance [40] (typically, the performance falls between that of CB-PWM and OPPs). For long prediction horizons, a modified sphere decoder has been proposed [41] to solve the underlying integer program, which has been successfully implemented on a field-programmable gate array [42].

However, (traditional) FCS-MPC does have some significant drawbacks. First, due to the absence of a pulse-width modulator, the harmonic spectrum of the switching signal is not well-defined. In fact, the harmonic spectrum is nondeterministic and includes harmonics at noninteger multiples

[3] For a three-phase three-level converter, the number of possible solutions is 3^{3N_p}.

of the fundamental frequency (that is, interharmonics exist). For applications with strict harmonic requirements (such as grid-connected converters), such a spectrum is not suitable. Second, a weighting factor has to be tuned (typically in a trial-and-error manner) in order to obtain a desired (average) switching frequency. Moreover, once the operating conditions change, the switching frequency also changes. Thus, the weighting factor has to be readjusted. Finally, it can also be observed in [40, Figure 9] that increasing the number of prediction steps does not always improve the harmonic performance. In order to obtain the optimal harmonic performance for a given prediction horizon, the weighting factor *and* sampling interval have to be adjusted (see [40, Figure 11]).

4.3 OPP-Based Control Techniques

OPP-based control techniques have significant advantages over FCS-MPC, since the (superior) steady-state harmonic performance is deterministic (the spectrum is determined by the OPP) and the device switching frequency can easily be adjusted via the pulse number. This section first motivates the benefits of OPPs. The underlying control difficulties of pulse patterns are also identified. This is followed by a brief review of some early OPP-based methods. Model predictive pulse pattern control, a modern OPP-based method that has seen usage in industry, is then reviewed and its primary limitation is highlighted. The section concludes with a review of OPP-based methods that attempt to address the limitation of model predictive pulse pattern control.

4.3.1 Early Methods

As shown in Section 3.4.4, OPPs significantly outperform CB-PWM in terms of harmonic distortion at low switching frequencies. This results in an OPP-modulated converter that is able to operate at low switching frequencies (which in turn leads to lower losses) while still having acceptable harmonic distortion. This is highly beneficial for medium-voltage applications, where the switching losses are high, since the power rating of a power converter can increase.

However, addressing the control problem of a converter system that is modulated with pulse patterns instead of CB-PWM is substantially more difficult. It is important to realize that OPPs are *optimal in steady-state conditions*; during transients, OPPs are suboptimal and lead to a slow response. This is exacerbated by a low switching frequency. A useful feature of CB-PWM is the existence of regularly-spaced sampling instants (which are at the peak and trough of the carrier signal) where the ripple of (certain) converter quantities is zero; it is possible to sample only the fundamental component. This fixed modulation interval is ideal for linear controllers, as the fundamental component can easily be regulated along its (sinusoidal) reference. In contrast, pulse patterns do not possess a fixed modulation interval. Therefore, regularly-spaced sampling instants where the ripple of certain converter quantities is zero do not exist; the fundamental component and the ripple are sampled. The ripple, which is a natural characteristic of an OPP, will be interpreted as an error by a linear controller. This leads to the controller attempting to regulate the ripple to zero and thus modify the steady-state pulse pattern. This will sacrifice the optimality of an OPP during steady-state conditions and thus increase the harmonic distortion. In order to prevent this, the bandwidth of a linear controller should be low so that it does not react to the ripple. Unfortunately, as a consequence, the response becomes sluggish. Moreover, the discontinuities that exist in the switching angles (see Figure 3.7) create current excursions when the modulation index changes in the quasi-steady state. Due to the slow response of a linear controller, these discontinuities must be eliminated. This requires additional restrictions (in the form of constraints) on the optimization problem underlying OPPs, which lead to pulse patterns with higher harmonic distortion during the steady state. In summary, linear controllers are not suitable for converter systems that are modulated

with pulse patterns.

As a first step away from a traditional linear controller for pulse patterns, a current trajectory controller was proposed in 1991 [43] for electrical machines. The method first determines the steady-state (stator) current trajectory that results from the pulse pattern. Then, during transients, a deadbeat-type controller regulates the stator current towards its steady-state trajectory by modifying the pulse pattern. In 2007, the method was adapted to control the stator flux instead of the stator current [44]. The advantage of tracking the stator flux trajectory is that the stator flux is independent of the leakage inductance of a machine. However, both methods require the fundamental component and ripple as separate quantities. Due to the fundamental component not being readily available when a converter is modulated by pulse patterns, a dedicated observer is required [45].

4.3.2 Model Predictive Pulse Pattern Control

Although MPC is a very active and popular research topic in the power electronics research community, there are few instances of MPC being readily used in industry. One exception is model predictive pulse pattern control (MP^{3}C), which is patented by ABB [46] and is being used in their modern drive systems [47]. Considered to be a state-of-the-art control technique, MP^{3}C formulates the stator flux trajectory control problem underlying pulse patterns in an MPC framework, resulting in a control technique with superior harmonic performance in steady-state conditions and a fast response during transients. Since the controller that is proposed in Chapter 5 is, in a sense, a generalization of MP^{3}C, the control principles underlying MP^{3}C are discussed in more detail than the early methods. For a full and complete overview on MP^{3}C, see [3] and [4, Chapter 12].

Consider an NPC converter that is modulated by a pulse pattern u and that is connected to the stator of an induction machine. For simplicity, only a single phase is considered (the concept generalizes to all three phases). When neglecting the stator resistance, the stator flux ψ_s of an induction machine is simply the integral of the (input) stator voltage,

$$\psi_s(t) = \psi_{s,0} + \frac{V_d}{2}\int_0^t u(\tau)\,\mathrm{d}\tau, \tag{4.1}$$

where $\psi_{s,0}$ is the stator flux at the initial (sampling) instant. With (4.1) as the internal dynamic model of the predictive controller, the stator flux is predicted at the *end* of the prediction horizon T_p [that is, $\psi_s(T_p)$ is predicted]. Denote with n_{sw} the number of switching transitions that occur during the prediction horizon T_p. It is fairly straightforward to show that by modifying the ith switching transition of a pulse pattern by $\Delta t_i = t_i - t_i^*$, where t_i and t_i^* are the modified and nominal switching times, respectively, that the modification in stator flux at the end of the horizon is

$$\Delta\psi_s(\Delta t_i) = -\frac{V_d}{2}\sum_{i=1}^{n_{\text{sw}}} \Delta t_i \Delta u_i, \tag{4.2}$$

where $\Delta u_i = u_i - u_{i-1} \in \{-1, 1\}$ is the change in switch position.

Similar to the early trajectory-based methods, MP^{3}C regulates the stator flux ψ_s along its (optimal) steady-state trajectory ψ_s^*. Since the stator flux is simply the (scaled) integral of the pulse pattern, it is easy to derive the steady-state trajectory.[4] In addition to reference tracking, and in accordance with the optimal control principle paradigm, the control effort (the switching time modification) is also penalized. The (quadratic) objective function underlying MP^{3}C is

$$J(\Delta t_i) = \left(\psi_{s,\text{err}} - \Delta\psi_s(\Delta t_i)\right)^2 + r\sum_{i=1}^{n_{\text{sw}}} \Delta t_i^2, \tag{4.3}$$

[4] Section 5.3 demonstrates how the steady-state trajectory for any load can be obtained.

where $\psi_{s,\text{err}} = \psi_{s,0} - \psi^*_{s,0}$ is the initial flux error. A positive penalty $r \in \mathbb{R}$ is required on the modifications. In the case that r is zero, MP^{3}C turns into a deadbeat-type controller and attempts to remove the flux error promptly.[5] Note that MP^{3}C minimizes the flux error at the end of the prediction horizon (and not across it). To maintain a feasible pulse pattern, the modified switching instants (in a particular phase) are not allowed to move into the past and must be in an ascending order, leading to the following $n_{\text{sw}} + 1$ constraints:

$$0 \leq t_1 \leq t_2 \leq \cdots \leq t_{n_{\text{sw}}} \leq t_{n_{\text{sw}}+1}. \tag{4.4}$$

The minimization of the objective function of (4.3) subject to the constraints of (4.4) constitutes a QP. Once the (optimal) modified switching instants have been determined, only the switching transitions within the sampling interval T_s (where T_s is typically much smaller than T_p) are applied. At the next sampling instant, the pulse pattern is re-optimized in accordance with the receding horizon policy.

When compared to the early methods, MP^{3}C has two major advantages. First, MP^{3}C considers the instantaneous error; the fundamental component and ripple are considered as a single quantity. Unlike the early methods that required the fundamental component and ripple separately, MP^{3}C does not require an additional observer to extract the fundamental component. Second, MP^{3}C is formulated in an MPC framework. The early methods were deadbeat-type controllers (which had no notion of a prediction horizon) that required a relatively coarse sampling interval in order to consider multiple switching transitions; all of the controller actions (modifications to the pulse pattern) over the (coarse) sampling interval were applied. In contrast, MP^{3}C has a short sampling interval and long prediction horizon, and utilizes the receding horizon principle. This inherently makes MP^{3}C more robust to noise and measurements errors.

Limitation

Although MP^{3}C is a state-of-the-art control technique that significantly outperforms conventional control methods used in industry (see [4, Chapter 13]), it is limited in its application. As observed from (4.1), the internal dynamic model is an integrator. Thus, MP^{3}C cannot capture the behaviour of higher-order systems, such as the grid-connected converter with an LC filter of Section 3.3.2; MP^{3}C is not applicable to any MIMO system. However, MP^{3}C can, to an extent, account for resonant behaviour of a higher-order system by adding an additional damping term in the objective function [48, 49], or by adding an outer dampening loop [50]. Although these approaches can be effective during steady-state operation, the performance during transients is rather poor. Specifically, in order to avoid exciting the resonance, large reference steps are filtered by a ramp-limiter, leading to a sluggish response.

4.3.3 OPP-Based Methods for Higher-Order Systems

As mentioned in Section 4.3.2, the internal dynamic model of MP^{3}C limits the method to first-order systems (more specifically, a pure integrator). For higher-order medium-voltage applications, a different control method is required that can operate at low switching frequencies with a good harmonic performance while having a short response time. However, very little research has been conducted on OPP-based controllers for a general MIMO system, making it a largely unexplored topic. Although there are existing control methods that attempt to address this problem, these methods do have some characteristics that either make them impractical or limit their performance.

[5] Refer to [3, Section 5-B] for more details on the deadbeat-based MP^{3}C algorithm.

In [51], an MPC control problem is formulated in the continuous-time domain. The modifications to the switching instants are determined by solving a nonlinear program. However, sinusoidal references are used instead of the (optimal) steady-state trajectory resulting from the pulse pattern. This implies that the (natural) ripple from the pulse pattern is interpreted as an error and will thus be modified, which during steady-state conditions results in suboptimal performance. The optimization problem, in its original nonlinear form, requires the gradient projection method to evaluate the matrix exponential at *every* iteration and, in addition, also requires a moderate amount of matrix multiplications; this results in a high computational burden. In order to simplify the optimization problem to a QP, the state variables are linearized around the nominal switching instants. However, the linearization requires that the three-phase voltage vectors have a fixed switching sequence. This implies that a switching transition in one phase cannot be shifted beyond a switching transition in another phase; thus imposing a major limitation. In combination with this limitation, the modifications to the switching transitions are also required to be very small due to the limited accuracy of the linearization. This results in very conservative constraints that lead to a sluggish response during transients. Furthermore, at certain operating points, the linearized problem also significantly increases the harmonic distortion.

In [52], the MPC control problem is formulated in the discrete-time domain. Relative to the current sampling instant k, the controller predicts the system states at even-indexed switching instants (that is, at $k+2$, $k+4$, and so forth) and can manipulate the odd-indexed switching instants (that is, at $k+1$, $k+3$, and so forth). Note that the prediction instants are, therefore, irregularly spaced. The method introduces a nonlinear transformation of the decision variables (which are the odd-indexed switching instants), transforming the problem from a nonconvex optimization problem into a QP. As with [51], the state references are also sinusoidal. However, the converter current (which has significant ripple) is slightly penalized, whereas the grid current (which is nearly sinusoidal and therefore matches its reference in the steady state) is penalized significantly. This choice of penalties results in the superior harmonic distortion of the OPP not being significantly worsened during the steady state. Although the method assumes a fixed switching order, the response time is still modest. However, the method does make an assumption regarding the characteristics of the load: the total energy in the system stays constant during an unforced response (that is, when the converter output voltage and the grid voltage are zero), implying elements that dissipate power have to be ignored in the internal model.

The latest attempt in a generalized OPP-based controller is proposed in [53, Chapter 4]. The MPC problem is formulated in the discrete-time domain at regularly-spaced prediction instants. The core principle of the method is to transform the discrete-valued pulse pattern into a real-valued control signal that is easy to modify. This is achieved by averaging the pulse pattern between the discrete-time instants. Once the real-valued signal is modified by solving a QP, a reverse transformation is required in order to retrieve the modified switching transitions. Unlike the methods in [51] and [52], this method uses the optimal steady-state trajectory that results from the pulse pattern as a reference. However, the method does have some limitations. Switching transitions are not allowed to move out of the sampling interval they are contained in, therefore restricting the degree that the pulse pattern can be modified. Thus, in order to allow moderate modifications, the sampling interval is required to be relatively coarse. On the other hand, the averaging step introduces an error in the steady-state pulse pattern, which increases as the sampling interval is increased. This creates conflicting objectives. The reverse transformation itself requires an additional optimization problem to be solved: a linear program. Nonetheless, the method results in a decent response time and holds the most promise when compared to [51] and [52].

The controller proposed in Chapter 5 has none of the limitations of the aforementioned control methods, and is a natural generalization of MP^{3}C. Furthermore, the proposed controller is practi-

cally viable, as it is implemented on a low-cost field-programmable gate array in Chapter 8 and can be executed in real-time; none of the aforementioned methods have been verified to be practically feasible.

Part II

Generalized Model Predictive Pulse Pattern Control Based on Small-Signal Modelling

Chapter 5

The Small-Signal Controller

In this chapter, the primary contribution of this thesis is presented: a generalized model predictive pulse pattern controller based on small-signal modelling. The chapter first describes the requirements of the control method. The notion of small-signal modelling and its use in the context of the proposed control method follows. A method is described that efficiently calculates the steady-state trajectory of a converter system that is modulated by a pulse pattern. The linearization of the modifications to a pulse pattern is then explained. Finally, a model predictive controller, which utilizes small-signal modelling, is derived. The performance of the controller during transients is evaluated via simulations, and the results are discussed.

Chapter Contents

5.1 Control Method Requirements

The controller proposed in this chapter is a generalized model predictive pulse pattern controller that is applicable to *any* linear MIMO converter system that is modulated by OPPs, overcoming the primary limitation of MP^{3}C mentioned in Section 4.3.2. When compared to the existing OPP-based control methods reviewed in Section 4.3.3, the proposed controller is a natural generalization of MP^{3}C and has no restrictions on modifications to a pulse pattern (except that the switching transitions must be nonnegative and occur in ascending order in each phase).

The formulation of the controller in Section 5.5 is general enough so that any converter topology can be considered. This includes inverters and rectifiers, single-phase and multi-phase systems, two-level and multi-level converters, voltage-source and current-source converters, and converters connected to electrical machines and to the grid. The only assumption is that the converter system must only contain linear elements such as ideal resistors, inductors, and capacitors. Furthermore, only linear quantities are directly controlled such as voltage, current, and flux.[1,2] Such a converter system can be described by a continuous-time state-space representation,

$$\frac{\mathrm{d}\boldsymbol{x}(t)}{\mathrm{d}t} = \boldsymbol{F}\boldsymbol{x}(t) + \boldsymbol{G}\boldsymbol{u}(t) + \boldsymbol{P}\boldsymbol{v}_g(t), \tag{5.1}$$

where $\boldsymbol{x} \in \mathbb{R}^{n_x}$ and $\boldsymbol{v}_g \in \mathbb{R}^{n_v}$ are the state and disturbance vectors, respectively. Note the it is assumed that the disturbance is known. The input vector $\boldsymbol{u} \in \mathbb{Z}^{n_u}$ describes the switch positions of the system, and is equal to the pulse pattern when the converter is modulated by OPPs. The state $\boldsymbol{F} \in \mathbb{R}^{n_x \times n_x}$, input $\boldsymbol{G} \in \mathbb{R}^{n_x \times n_u}$, and disturbance $\boldsymbol{P} \in \mathbb{R}^{n_x \times n_v}$ matrices characterize the system. It is assumed that all parameters are exactly known.

The control method proposed in this chapter regulates the state vector $\boldsymbol{x}$ of a linear MIMO converter system along a suitable reference trajectory $\boldsymbol{x}^*$ by

- modifying the nominal pulse pattern during transients to achieve a short response time, and
- modulating the converter with the nominal pulse pattern during the steady state to achieve the superior harmonic performance of OPPs.

As a case study to demonstrate the controller, the grid-connected NPC converter system from Section 3.3.2 is considered. The control problem underlying grid-connected converters is, typically, to control the real power P and reactive power Q to their respective references P^* and Q^*. Although, strictly speaking, the power should be controlled at the point of common coupling, the power is instead controlled directly at the grid voltage source for convenience. The state vector is the converter currents, grid currents, and capacitor voltages in the $\alpha\beta$ reference frame,

$$\boldsymbol{x}(t) = \begin{bmatrix} i_\alpha(t) & i_\beta(t) & i_{g,\alpha}(t) & i_{g,\beta}(t) & v_{c,\alpha}(t) & v_{c,\beta}(t) \end{bmatrix}^{\mathrm{T}}$$

and the disturbance vector is the grid voltage in the $\alpha\beta$ reference frame,

$$\boldsymbol{v}_g(t) = \begin{bmatrix} v_{g,\alpha}(t) \\ v_{g,\beta}(t) \end{bmatrix} = \sqrt{\frac{2}{3}} V_{g,\mathrm{LL}} \begin{bmatrix} \sin(\omega_1 t) \\ -\cos(\omega_1 t) \end{bmatrix}.$$

[1] Quantities such as power and electromechanical torque are products of two system states (hence, nonlinear) and are not directly controlled. However, these quantities can be controlled by having an outer control loop generate suitable references for the system states.

[2] In Chapter 7, the controller is extended to control the neutral-point potential, which is a product of states and inputs.

From the differential equations of (3.15) that describe the converter system, the state-space matrices follow as

$$\boldsymbol{F} = \begin{bmatrix} -\frac{R+R_C}{L}\mathbf{I}_2 & \frac{R_C}{L}\mathbf{I}_2 & -\frac{1}{L}\mathbf{I}_2 \\ \frac{R_C}{L_{gt}}\mathbf{I}_2 & -\frac{R_{gt}+R_C}{L_{gt}}\mathbf{I}_2 & \frac{1}{L_{gt}}\mathbf{I}_2 \\ \frac{1}{C}\mathbf{I}_2 & -\frac{1}{C}\mathbf{I}_2 & \mathbf{0}_{2\times 2} \end{bmatrix}, \ \boldsymbol{G} = \frac{V_d}{2L}\begin{bmatrix} \mathbf{I}_2 \\ \mathbf{0}_{2\times 2} \\ \mathbf{0}_{2\times 2} \end{bmatrix} \boldsymbol{K}, \text{ and } \boldsymbol{P} = \frac{1}{L_{gt}}\begin{bmatrix} \mathbf{0}_{2\times 2} \\ -\mathbf{I}_2 \\ \mathbf{0}_{2\times 2} \end{bmatrix}.$$

The input to the system is a three-phase three-level pulse pattern $\boldsymbol{u}_{abc} \in \{-1, 0, 1\}^3$,

$$\boldsymbol{u}(t) = \boldsymbol{u}_{abc}(t) = \begin{bmatrix} u_a(t) & u_b(t) & u_c(t) \end{bmatrix}^{\mathrm{T}}.$$

Note that all variables, except for the input, are referred to the $\alpha\beta$ reference frame.

5.2 Overview of Small-Signal Modelling

Although no clear definition of *small-signal modelling* could be found in literature, the term, usually, refers to a linearization of a nonlinear system around an operating point [54, Section 3.10]. The term is exclusively used in electrical engineering, and there are many uses and methods of small-signal modelling. A common usage is to obtain a linear approximation of semiconductor devices [55]. In power electronics, small-signal modelling has been used to obtain an accurate linear model of pulse-width modulators [56], from which the stability of the system can be accurately analyzed. Note that small-signal modelling does not strictly imply a linearization around a single operating point; the system can also be linearized around a trajectory (which is the case in this thesis). In this thesis, small-signal modelling is used to obtain a linear model for the modifications to a pulse pattern; for higher-order systems, (nonlinearized) modifications to the switching instants of the pulse pattern have a nonlinear effect on the system states as the modifications will be arguments of matrix exponentials.

There are three types of variables associated with small-signal modelling. The first is the *complete-signal variables*

$$\xi(t) = \xi^*(t) + \tilde{\xi}(t), \tag{5.2}$$

which is the sum of the *large-signal variables* ξ^* and *small-signal variables* $\tilde{\xi}$. The large-signal variables represent the trajectory that the system is linearized around. The small-signal quantities represent the linearized variables, which are perturbations superimposed on the large-signal variables. In this thesis, the large-signal variables are the steady-state variables (the nominal pulse pattern and its resulting steady-state trajectory), which are interpreted as references. The complete-signal variables represent the actual (measured) values of the system. From (5.2), the small-signal variables can be expressed as

$$\tilde{\xi}(t) = \xi^*(t) - \xi(t), \tag{5.3}$$

which is interpreted as an error (specifically, small-signal state variables represent errors). A controller can be used to modify the small-signal input in order to drive the small-signal state variables to zero. In steady-state conditions, all small-signal variables are zero.

It is shown in Section 5.4 that modifications to a pulse pattern can be *approximated* by the strengths of impulses (which are the small-signal input). This results in the switching instant modifications to the pulse pattern being removed from the matrix exponentials, and the small-signal state variables (the error) being linear in the impulse strengths. The model predictive controller developed in Section 5.5 then manipulates the impulse strengths in order to drive the state vector onto its steady-state trajectory. First, the steady-state trajectory resulting from the nominal pulse pattern is required, which is derived in the next section.

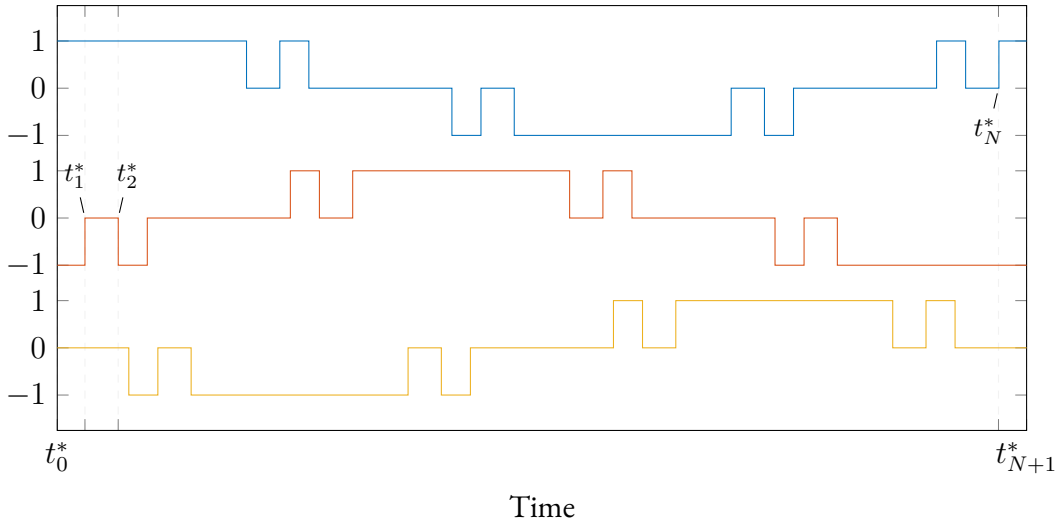


Figure 5.1: Three-phase nominal pulse pattern $\boldsymbol{u}_{abc}^*$. The pulse number is $d = 3$, and there are a total of $N = 36$ switching transitions across all three phases.

5.3 Steady-State Trajectory of a Converter System

The following method is a modification to the method proposed in [57].

It can be shown from (5.1) that the steady-state trajectory resulting from the nominal pulse pattern $\boldsymbol{u}_{abc}^*$ at time $t \in [0, T_1]$, where T_1 is the fundamental period, is

$$\boldsymbol{x}^*(t) = \boldsymbol{e}^{\boldsymbol{F}t}\boldsymbol{x}_0^* + \int_0^t \boldsymbol{e}^{\boldsymbol{F}(t-\tau)}\boldsymbol{G}\boldsymbol{u}_{abc}^*(\tau)\,\mathrm{d}\tau + \int_0^t \boldsymbol{e}^{\boldsymbol{F}(t-\tau)}\boldsymbol{P}\boldsymbol{v}_g(\tau)\,\mathrm{d}\tau, \tag{5.4}$$

where $\boldsymbol{x}_0^*$ is the initial steady-state value at the start of the fundamental period T_1. In order to calculate the steady-state trajectory over a fundamental period, the initial steady-state value $\boldsymbol{x}_0^*$ first needs to be calculated. Thanks to linearity, the principle of superposition can be used to calculate the effect of the pulse pattern and grid voltage separately as

$$\boldsymbol{x}^*(t) = \boldsymbol{x}_{\text{OPP}}^*(t) + \boldsymbol{x}_g^*(t) \tag{5.5}$$

where

$$\boldsymbol{x}_{\text{OPP}}^*(t) = \boldsymbol{e}^{\boldsymbol{F}t}\boldsymbol{x}_{\text{OPP},0}^* + \int_0^t \boldsymbol{e}^{\boldsymbol{F}(t-\tau)}\boldsymbol{G}\boldsymbol{u}_{abc}^*(\tau)\,\mathrm{d}\tau. \tag{5.6}$$

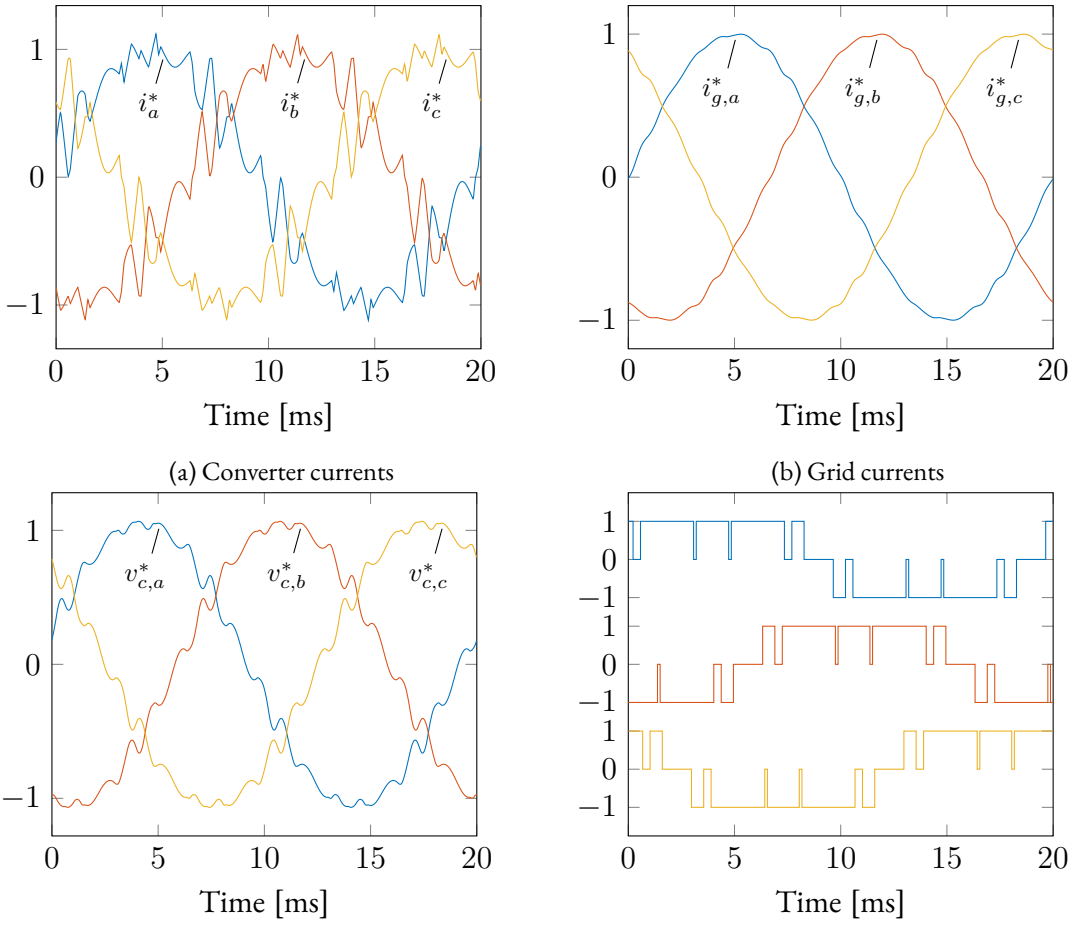
For now, the grid is ignored and assumed to be short-circuited. Consider the (three-phase) nominal pulse pattern in Figure 5.1, which has $N = 12d$ switching transitions over the fundamental period T_1. Denote with t_i^* the (three-phase) nominal switching times, where $i = 0, 1, \ldots, N+1$. The switching times $t_0^* = 0$ and $t_{N+1}^* = T_1$ are introduced to take the boundaries into account. Due to periodicity, it always holds that

$$\boldsymbol{x}_{\text{OPP}}^*(0) = \boldsymbol{x}_{\text{OPP}}^*(T_1) \tag{5.7}$$

in steady-state conditions; this fact can be used to calculate $\boldsymbol{x}_{\text{OPP},0}^*$. It is easy to see from (5.6) that

$$\boldsymbol{x}_{\text{OPP}}^*(0) = \mathbf{I}_{n_x}\boldsymbol{x}_{\text{OPP},0}^*, \tag{5.8}$$

where $\mathbf{I}_{n_x}$ is the identity matrix of dimensions n_x. To calculate $\boldsymbol{x}_{\text{OPP}}^*(T_1)$, note that the three-phase pulse pattern is constant between switching transitions. This leads to the integral of (5.6) (with



(a) Converter currents

(b) Grid currents

(c) Capacitor voltages

(d) Switch positions

Figure 5.2: The steady-state trajectory $\boldsymbol{x}^*$ (in pu) of the converter system resulting from the nominal pulse pattern $\boldsymbol{u}_{abc}^*$.

$t = T_1$) becoming

$$\begin{aligned}\boldsymbol{f} &= \sum_{i=0}^{i=N} \int_{t_i^*}^{t_{i+1}^*} \boldsymbol{e}^{\boldsymbol{F}(T_1-\tau)}\, \mathrm{d}\tau\, \boldsymbol{G}\boldsymbol{u}_{abc}^*(t_i^*) \\ &= \boldsymbol{F}^{-1} \sum_{i=0}^{i=N} \left((\boldsymbol{e}^{\boldsymbol{F}(T_1-t_i^*)} - \boldsymbol{e}^{\boldsymbol{F}(T_1-t_{i+1}^*)}) \boldsymbol{G}\boldsymbol{u}_{abc}^*(t_i^*) \right),\end{aligned}$$

where $\boldsymbol{F}$ is assumed to be invertible. Thus, the steady-state value at the end of the fundamental period is

$$\boldsymbol{x}_{\mathrm{OPP}}^*(T_1) = \boldsymbol{e}^{\boldsymbol{F}T_1}\boldsymbol{x}_{\mathrm{OPP},0}^* + \boldsymbol{f}. \tag{5.9}$$

It is easy to show from (5.7) [after inserting (5.8) and (5.9)] that the initial steady-state value resulting from the pulse pattern is

$$\boldsymbol{x}_{\mathrm{OPP},0}^* = (\mathbf{I}_{n_x} - \boldsymbol{e}^{\boldsymbol{F}T_1})^{-1}\boldsymbol{f}, \tag{5.10}$$

where $\boldsymbol{x}_{\mathrm{OPP},0}$ is assumed to be unique in the steady state.[3]

[3]This is true for most plants. However, for plants where the resistive components are not taken into account (such as an integrator or resonator), there is no unique trajectory since $\boldsymbol{e}^{\boldsymbol{F}t}$ will not diminish as $t \to \infty$.

With the initial steady-state value $\boldsymbol{x}^*_{\text{OPP},0}$ known, the steady-state trajectory $\boldsymbol{x}^*_{\text{OPP}}$ resulting from the pulse pattern can be calculated over the fundamental period using (5.6). Typically, the steady-state trajectory $\boldsymbol{x}^*_{\text{OPP}}$ is required at discrete sampling instants of T_s (the sampling interval of the controller). Alternatively, instead of the continuous-time representation of (5.6), a discrete-time representation can be used to calculate the steady-state trajectory over a fundamental period, although there will be a slight error due to the quantization of the switching instants of the pulse pattern.

Now, consider the grid voltage. The steady-state trajectory resulting from the grid voltage can be described as

$$\boldsymbol{x}^*_g(t) = \boldsymbol{e}^{\boldsymbol{F}t}\boldsymbol{x}^*_{g,0} + \int_0^t \boldsymbol{e}^{\boldsymbol{F}(t-\tau)}\boldsymbol{P}\boldsymbol{v}_g(\tau)\,\mathrm{d}\tau,$$

where the integral is not trivial to solve since the grid voltage is sinusoidal. Fortunately, the effect of the grid can be calculated using well-known phasor analysis; all the converter states resulting from the grid voltage only consist of a fundamental component. After calculating the effect of the grid voltage over the fundamental period, the steady-state trajectory $\boldsymbol{x}^*_g$ resulting from the grid voltage can be added to that of the pulse pattern as shown with (5.5). Alternatively, the effect of the grid voltage can be considered by representing the grid voltage as a harmonic resonator and including it as additional state variables (this approach is used in Section 7.2).

Figure 5.2 shows an example of a steady-state trajectory $\boldsymbol{x}^*$ in the abc reference frame. The pulse number is $d = 5$ and the system is operating at rated conditions with unity power factor.

5.4 Modelling Modifications of a Pulse Pattern

This section demonstrates how the modifications to a pulse pattern are modelled. First, in Section 5.4.1, it is shown that the modifications can be approximated by using the strengths of impulses. This results in the modifications to the state vector being described by a linear set of differential equations during the design of a model predictive controller in Section 5.5.1. The major benefit of this linearization is that the underlying optimization problem, which yields the optimal impulse strengths (representing the optimal modifications), becomes a QP instead of a significantly more complex nonlinear program involving the matrix exponential. However, the approximation of using impulses does introduce a margin of error during the predicted system response. To improve the accuracy, the pulse pattern is successively linearized around the modified switching instants as described in Section 5.4.2.

Before continuing, three types of pulse patterns are introduced. For now, only a single-phase pulse pattern is considered. The first type is called the *nominal pulse pattern*, which is described as

$$u^*(t) = u^*_0 + \sum_{i=1}^{n^*} \Delta u^*_i h(t - t^*_i), \tag{5.11}$$

where $h(t)$ is the (unit) step function, $u^*_0 \in \{-1, 0, 1\}$ is the initial nominal switch position, and $\Delta u^*_i \in \{-1, 1\}$ is the ith switching transition direction. The nominal pulse pattern is the offline-calculated pulse pattern that is optimal in steady-state conditions. The (nominal) pulse pattern has n^* switching transitions that occur at the *nominal* switching instants t^*_i.

The second type is the *incumbent pulse pattern*, which has n switching transitions that occur at the *incumbent* switching instants t'_i,

$$u(t) = u_0 + \sum_{i=1}^{n} \Delta u_i h(t - t'_i), \tag{5.12}$$

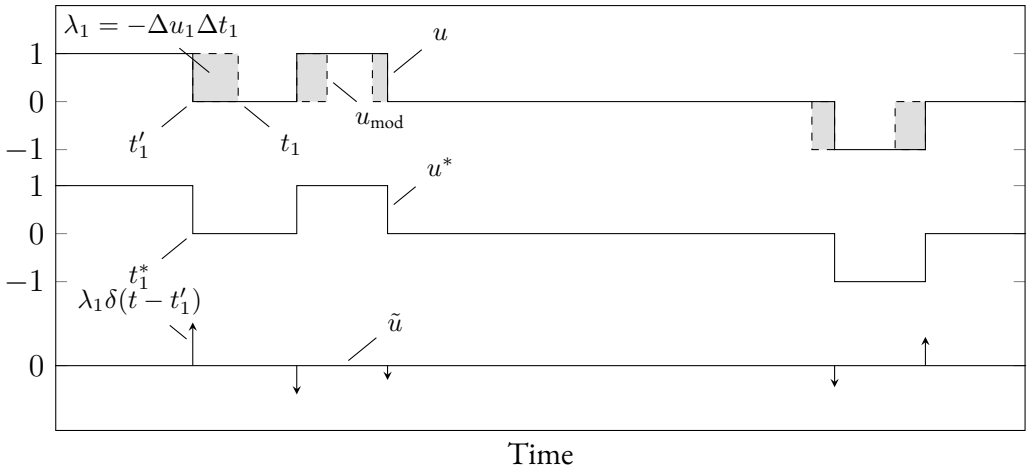


Figure 5.3: Using impulses to represent rectangular pulses (that is, modifications to a pulse pattern).

where u_0 and Δu_i are the initial switch position and ith switching transition, respectively.

The third and final type of pulse pattern is the *modified pulse pattern*, which results from modifications to the incumbent pulse pattern. The pulse pattern has switching transitions that occur at the *modified* switching instants $t_i = t_i' + \Delta t_i$, and is described as

$$u_{\text{mod}}(t) = u_0 + \sum_{i=1}^{n} \Delta u_i h(t - (t_i' + \Delta t_i)). \tag{5.13}$$

In summary, the nominal pulse pattern u^* is the offline-calculated pulse pattern (which is never modified and is part of the large-signal model), the incumbent pulse pattern u is the pulse pattern that is used as a starting point to optimize the switching instants, and the modified pulse pattern u_{mod} is the pulse pattern that results from the modifications to the incumbent pulse pattern (and is part of the complete-signal model).

5.4.1 Linear Approximation to Modifications of a Pulse Pattern

Consider the modified pulse pattern u_{mod}. It can be observed in Figure 5.3 that by modifying the ith switching transition by Δt_i, a rectangular pulse with an area of

$$\lambda_i = -\Delta u_i \Delta t_i \tag{5.14}$$

is added or removed to the incumbent pulse pattern u. It is known that an impulse with strength λ_i can be used to approximate the area λ_i of narrow rectangular pulses (this fact is used in [56]). Specifically, their (definite) integrals are the same. To further motivate this, consider a first-order Taylor series expansion of the step functions of the modified pulse pattern u_{mod} of (5.13) around the incumbent switching instants t_i',[4]

$$u_{\text{mod}}(t) \approx u_0 + \sum_{i=1}^{n} \Delta u_i \left(h(t - t_i') - \Delta t_i \delta(t - t_i') \right),$$

which can be written as[5]

$$u_{\text{mod}}(t) = u_0 + \sum_{i=1}^{n} \Delta u_i h(t - t_i') + \sum_{i=1}^{n} \lambda_i \delta(t - t_i') \tag{5.15}$$

[4] A first-order Taylor series expansion is $f(t + \Delta t) \approx f(t) + \Delta t \frac{\mathrm{d}f(t)}{\mathrm{d}t}$.

[5] For convenience, the approximation is replaced with an equality in the sequel.

with the definition of (5.14). As expected, the (linearized) modified pulse pattern u_{mod} consists of the incumbent pulse pattern u and the (approximated) modifications represented by impulses with strengths λ_i. Specifically, the strength λ_i of the ith impulse represents the ith (rectangular) area.

For the moment, assume that the incumbent pulse pattern u and nominal pulse pattern u^* are equal (as is the case in Figure 5.3). Then, according to the definition of small-signal variables in (5.3), the *small-signal input* is defined as

$$\begin{aligned}\tilde{u}(t) &= u_{\text{mod}}(t) - u^*(t) \\ &= \sum_{i=1}^{n} \lambda_i \delta(t - t_i').\end{aligned} \tag{5.16}$$

The small-signal input $\tilde{u}$ consists of impulses with strengths λ_i that occur at the incumbent switching instants t_i'.

The model predictive controller developed in Section 5.5 adjusts the impulse strengths in order to drive the system into the steady state. After the (optimal) impulse strengths are calculated by the controller, they can be translated into time modifications as

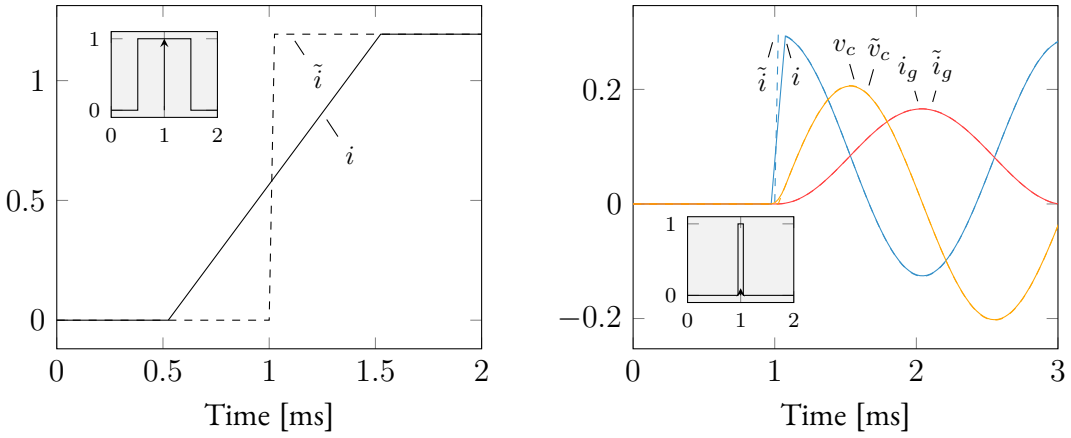
$$\Delta t_i = -\frac{\lambda_i}{\Delta u_i}, \tag{5.17}$$

which follows from (5.14). These modifications can then be added to the incumbent switching times to yield the modified switching times, $t_i = t_i' + \Delta t_i$.

Accuracy of Approximation

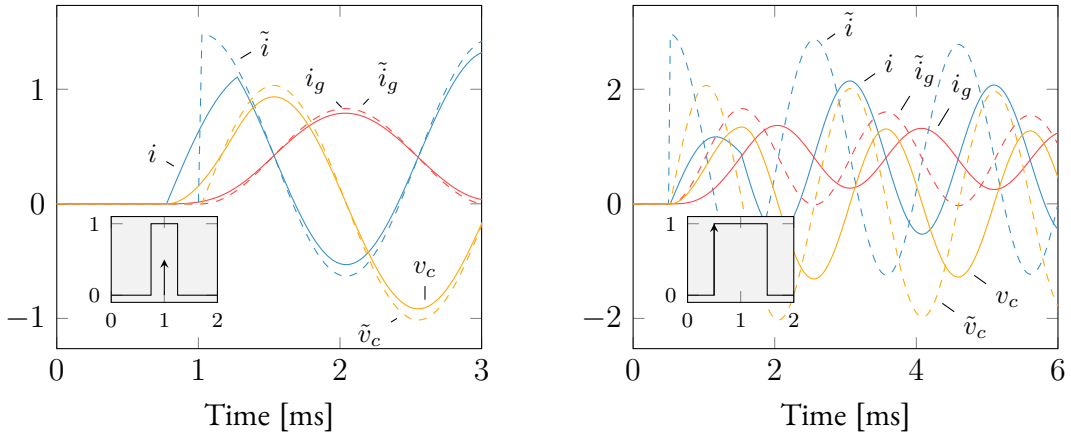
It is stressed that the impulses *approximate* the rectangular areas; the narrower the rectangular pulses, the better the approximation. Furthermore, the placement of the impulses also has an effect; ideally, the impulses should be placed at the center of the rectangle. However, since the widths of the rectangles are not known in advance, it is not possible to do so without rendering the linearization pointless. Instead, the impulses are simply placed at the incumbent switching instants (as derived from the first-order Taylor series expansion). Finally, the characteristics of the load also determine how well the strength of an impulse can represent a rectangular area.

To demonstrate the aforementioned statements, consider the responses shown in Figure 5.4. For convenience, only single-phase systems are used. The state variables that result from the impulse are denoted with $\tilde{\xi}$, whereas those that result from a rectangular pulse (with its area equal to the impulse strength) are denoted with ξ. In Figure 5.4a, the parameters from Table 3.5 are used with the resistance neglected (thus, the system is a pure integrator). The load voltage source is also set to zero. The impulse is placed at the center of the rectangle. It can be observed that an impulse is a (near) perfect approximation of the rectangle, with the only difference being during the integration period. This observation holds true no matter the width of the rectangle. However, this case is an exception and there will be a margin of error in the approximation for other systems. In Figures 5.4b–5.4d, the parameters of the grid-connected converter system from Section 3.3.2 are used (with the grid voltage set to zero), which constitutes a third-order system. In Figure 5.4b, a narrow rectangular pulse with a width of $\Delta t = 0.1\,\text{ms}$ is applied to the system. It can be observed that the (barely noticeably) impulse is a good approximation, with the responses resulting from the rectangle and impulse being near identical. The width of the rectangular pulse is increased to $\Delta t = 0.5\,\text{ms}$ in Figure 5.4c. Now, the margin of error is noticeable in the responses. Nonetheless, the approximation is decent. The width is further increased to $\Delta t = 1\,\text{ms}$ and the impulse is now placed at the start of the rectangle in Figure 5.4d. It is obvious that the impulse poorly approximates the larger rectangular pulse. In



(a) Integrator. The impulse is placed at the center of a rectangle with width $\Delta t = 1$ ms.

(b) Third-order load. The impulse is placed at the center of the rectangle with width $\Delta t = 0.1$ ms.



(c) Third-order load. The impulse is placed at the center of the rectangle with width $\Delta t = 0.5$ ms.

(d) Third-order load. The impulse placed is at the start of the rectangle with width $\Delta t = 1$ ms.

Figure 5.4: Responses (in pu) of an impulse and its equivalent rectangular pulse. The impulse strengths are scaled up by a factor of 1000 in order to be visible.

addition, since the impulse is not placed at the center of the rectangular pulse, there is a time shift of $\frac{\Delta t}{2}$ (half the pulse width) present that only worsens the approximation.

The next section proposes a relatively simple technique in order to overcome these inaccuracies and improve the internal model of the controller.

5.4.2 Enabling Accurate Predictions of Modifications of a Pulse Pattern

As shown in Section 5.4.1, the strength of impulses can be used to approximate the time modifications to the switching instants of a pulse pattern. The primary benefit of this is that the underlying optimization problem in Section 5.5.4 is a QP. However, as illustrated with Figure 5.4, only small modifications can be accurately represented by impulses; large modifications can lead to severe inaccuracies. Fortunately, it is possible to accurately predict the effect of the actual rectangular pulses *after* the modified switching instants t_i have been calculated. This is achieved by updating the incumbent switching times t_i' at the *current* instant with the modified switching instants t_i calculated at the *previous* instant; this is illustrated with Figure 5.5. Here, the incumbent switching times t_i' are equal to the modified switching times t_i of Figure 5.3. In this case, with slight abuse of notation,

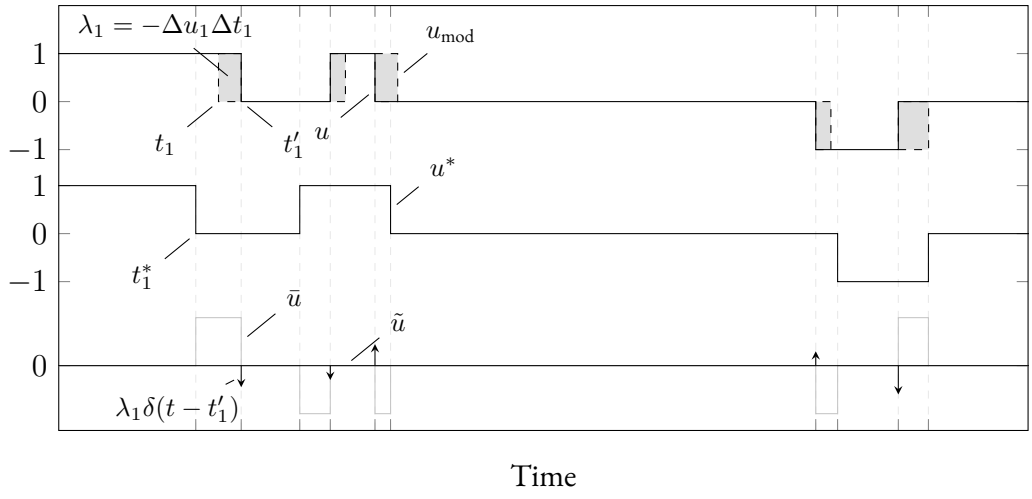


Figure 5.5: Illustration of how previously-calculated impulse strengths are represented by actual rectangles.

the difference between the modified pulse pattern and the nominal pulse pattern yields[6]

$$u_{\text{mod}}(t) - u^*(t) = \tilde{u}(t) + \bar{u}(t),$$

where

$$\begin{aligned} \bar{u}(t) &= u(t) - u^*(t) \\ &= (u_0 - u_0^*) + \left(\sum_{i=1}^{n} \Delta u_i h(t - t_i') - \sum_{i=1}^{n^*} \Delta u_i^* h(t - t_i^*) \right) \end{aligned} \tag{5.18}$$

is defined as the *rectangle input* and represents the actual rectangular pulses of previous impulse strengths. This enables a (predictive) controller to accurately predict the result of its (previous) corrections and, if required, adjust them accordingly.

5.4.3 Three-Phase Case

This section generalizes the notation and techniques developed in the previous sections to the three-phase case. The modification of the ith switching instant of phase $p \in \{a, b, c\}$ is

$$\Delta t_{p,i} = t_{p,i} - t'_{p,i}, \tag{5.19}$$

where $t_{p,i}$ and $t'_{p,i}$ are the modified and incumbent switching instants, respectively. The (direction of the) ith switching transition in a particular phase is

$$\Delta u_{p,i} = u_{p,i} - u_{p,i-1},$$

where $u_{p,i}, u_{p,i-1} \in \{-1, 0, 1\}$. By generalizing (5.14), the impulse strength associated with the ith switching transition in a particular phase is

$$\lambda_{p,i} = -\Delta u_{p,i} \Delta t_{p,i}. \tag{5.20}$$

Consider the three-phase modified pulse pattern over the interval $t \in [0, T_p]$, where T_p is the *prediction horizon*. According to the principle developed in Section 5.4.1 of using impulses to represent modifications, the (linearized) modified pulse pattern is

$$\boldsymbol{u}_{abc,\text{mod}}(t, \lambda_{p,i}) = \boldsymbol{u}_{abc}(t) + \tilde{\boldsymbol{u}}_{abc}(t, \lambda_{p,i}) \tag{5.21}$$

[6] According to (5.3), it should be written $u_{\text{mod}} - u^* = \tilde{u}$. However, $\tilde{u}$ is reserved for the impulses as in (5.16). To account for the additional terms, $\bar{u}$ is introduced.

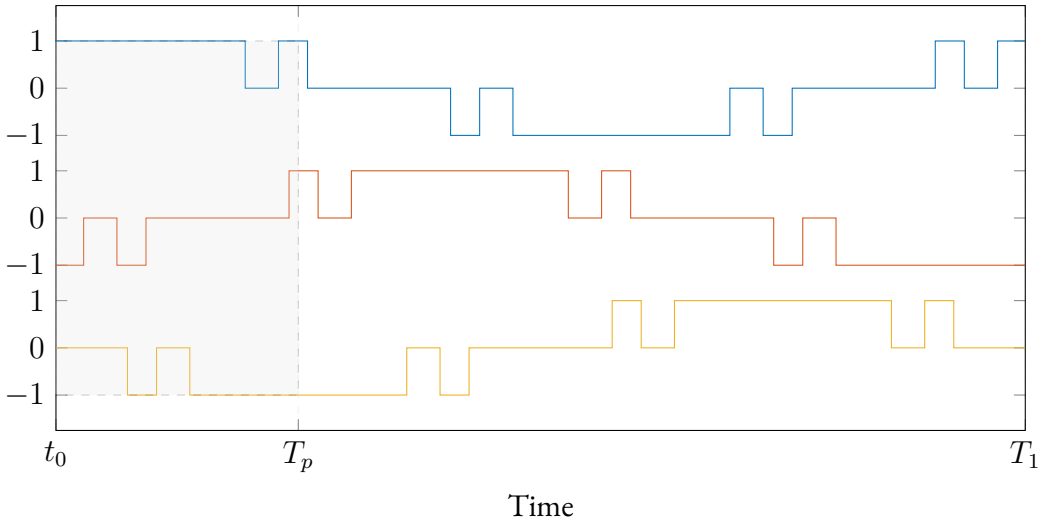


Figure 5.6: Switching transitions of a three-phase pulse pattern $\boldsymbol{u}_{abc}$ that fall within the prediction horizon T_p.

where

$$\boldsymbol{u}_{abc}(t) = \begin{bmatrix} u_a(t) \\ u_b(t) \\ u_c(t) \end{bmatrix} = \begin{bmatrix} u_{a,0} + \sum_{i=1}^{n_a} \Delta u_{a,i} h(t - t'_{a,i}) \\ u_{b,0} + \sum_{i=1}^{n_b} \Delta u_{b,i} h(t - t'_{b,i}) \\ u_{c,0} + \sum_{i=1}^{n_c} \Delta u_{c,i} h(t - t'_{c,i}) \end{bmatrix} \tag{5.22}$$

is the three-phase incumbent pulse pattern, and

$$\tilde{\boldsymbol{u}}_{abc}(t, \lambda_{p,i}) = \begin{bmatrix} \tilde{u}_a(t, \lambda_{a,i}) \\ \tilde{u}_b(t, \lambda_{b,i}) \\ \tilde{u}_c(t, \lambda_{c,i}) \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{n_a} \lambda_{a,i} \delta(t - t'_{a,i}) \\ \sum_{i=1}^{n_b} \lambda_{b,i} \delta(t - t'_{b,i}) \\ \sum_{i=1}^{n_c} \lambda_{c,i} \delta(t - t'_{c,i}) \end{bmatrix} \tag{5.23}$$

is the three-phase small-signal input (a weighted train of impulses). Here, n_p denotes the number of switching transitions in phase p that fall within the prediction horizon T_p. Note that the incumbent switching times are defined according to the current time instant $t_0 = 0$. The total number of switching transitions are denoted with $n_{\text{sw}} = n_a + n_b + n_c$. With Figure 5.6 as an example, the number of switching transitions that fall within the horizon in each of the three phases is $n_a = 2$, $n_b = 4$, and $n_c = 3$, whereas the total number of switching transitions is $n_{\text{sw}} = 9$.

Similarly, the nominal pulse pattern follows as

$$\boldsymbol{u}^*_{abc}(t) = \begin{bmatrix} u^*_a(t) \\ u^*_b(t) \\ u^*_c(t) \end{bmatrix} = \begin{bmatrix} u^*_{a,0} + \sum_{i=1}^{n^*_a} \Delta u^*_{a,i} h(t - t^*_{a,i}) \\ u^*_{b,0} + \sum_{i=1}^{n^*_b} \Delta u^*_{b,i} h(t - t^*_{b,i}) \\ u^*_{c,0} + \sum_{i=1}^{n^*_c} \Delta u^*_{c,i} h(t - t^*_{c,i}) \end{bmatrix}, \tag{5.24}$$

with n^*_p switching transitions in phase p occurring at the nominal switching instants $t^*_{p,i}$. The (so-called) three-phase rectangle pulse pattern follows from (5.18) as

$$\bar{\boldsymbol{u}}_{abc}(t) = \boldsymbol{u}_{abc}(t) - \boldsymbol{u}^*_{abc}(t). \tag{5.25}$$

5.5 The Small-Signal Controller

This section introduces the generalized model predictive pulse pattern controller, which is referred to as the *small-signal controller*. The controller is formulated according to the MPC framework. First, Section 5.5.1 derives the internal dynamic model of the controller, which describes the error

of the system state vector over the prediction horizon. The internal model of the controller utilizes impulse strengths to model corrections to the pulse pattern (as developed in Section 5.4). Using the internal dynamic model, an objective function is formulated in Section 5.5.2 that penalizes the tracking error and modifications to the pulse pattern. Constraints are introduced in Section 5.5.3 that maintain a feasible pulse pattern. The objective function and constraints constitute the optimization problem described in Section 5.5.4, which, when solved, yields the optimal impulse strengths that can be translated into time modifications. Finally, Section 5.5.5 explains and illustrates the receding horizon policy, which introduces feedback into the system. The control algorithm is explained in Section 5.5.6. A reduced version of the controller is explained in Section 5.5.7.

5.5.1 Internal Dynamic Model

Recall the state-space representation of (5.1) that describes the evolution of a converter system. By integrating (5.1), and with the modified pulse pattern $\boldsymbol{u}_{abc,\mathrm{mod}}$ as the input, the state vector can be predicted at the time $t \in [0, T_p]$ as

$$\boldsymbol{x}(t, \lambda_{p,i}) = \boldsymbol{e}^{\boldsymbol{F}t}\boldsymbol{x}_0 + \int_0^t \boldsymbol{e}^{\boldsymbol{F}(t-\tau)}\boldsymbol{G}\boldsymbol{u}_{abc,\mathrm{mod}}(\tau, \lambda_{p,i})\,\mathrm{d}\tau + \int_0^t \boldsymbol{e}^{\boldsymbol{F}(t-\tau)}\boldsymbol{P}\boldsymbol{v}_g(\tau)\,\mathrm{d}\tau, \tag{5.26}$$

where $\boldsymbol{x}_0$ is the initial state that is measured at $t_0 = 0$. Similarly, as shown in Section 5.3, the (optimal) steady-state trajectory follows from the nominal pulse pattern $\boldsymbol{u}^*_{abc}$ over the prediction horizon T_p as

$$\boldsymbol{x}^*(t) = \boldsymbol{e}^{\boldsymbol{F}t}\boldsymbol{x}^*_0 + \int_0^t \boldsymbol{e}^{\boldsymbol{F}(t-\tau)}\boldsymbol{G}\boldsymbol{u}^*_{abc}(\tau)\,\mathrm{d}\tau + \int_0^t \boldsymbol{e}^{\boldsymbol{F}(t-\tau)}\boldsymbol{P}\boldsymbol{v}_g(\tau)\,\mathrm{d}\tau, \tag{5.27}$$

where $\boldsymbol{x}^*_0$ is the initial optimal state.[7] According to (5.3), the small-signal state vector is defined as

$$\tilde{\boldsymbol{x}}(t, \lambda_{p,i}) = \boldsymbol{x}(t, \lambda_{p,i}) - \boldsymbol{x}^*(t), \tag{5.28}$$

which is referred to as the *small-signal error*; this term is used to emphasize that the small-signal state vector represents an error. By inserting (5.26) and (5.27) into (5.28), the small-signal error is described as

$$\tilde{\boldsymbol{x}}(t, \lambda_{p,i}) = \boldsymbol{e}^{\boldsymbol{F}t}\tilde{\boldsymbol{x}}_0 + \int_0^t \boldsymbol{e}^{\boldsymbol{F}(t-\tau)}\boldsymbol{G}\bar{\boldsymbol{u}}_{abc}(\tau)\,\mathrm{d}\tau + \int_0^t \boldsymbol{e}^{\boldsymbol{F}(t-\tau)}\boldsymbol{G}\tilde{\boldsymbol{u}}_{abc}(\tau, \lambda_{p,i})\,\mathrm{d}\tau, \tag{5.29}$$

where $\tilde{\boldsymbol{x}}_0 = \boldsymbol{x}_0 - \boldsymbol{x}^*_0$ is the initial small-signal error. It can be observed that the grid is not part of the small-signal model. However, the grid voltage is required when selecting the appropriate pulse pattern according to the operating conditions. Note that for an unmodified pulse pattern ($\tilde{\boldsymbol{u}}_{abc}$ and $\bar{\boldsymbol{u}}_{abc}$ are zero), the (unforced) small-signal error naturally diminishes over time; this is expected since the system will eventually enter the steady state, albeit extremely slowly since the time constants are typically long.

Observe that when the system is a pure integrator ($\boldsymbol{F} = \boldsymbol{0}_{n_x \times n_x}$), and when $\bar{\boldsymbol{u}}_{abc}$ is zero, that (5.29) reduces to the internal dynamic model of MP^{3}C in (4.1). Furthermore, $\lambda_{p,i} = -\Delta u_{p,i}\Delta t_{p,i}$ is contained in the correction term of MP^{3}C in (4.2). It can be concluded that the small-signal controller is a natural generalization of MP^{3}C to higher-order systems. However, MP^{3}C only considers the system states at the *end* of the prediction horizon, whereas the small-signal controller considers the systems states *across* the prediction horizon.

[7] Note that $\boldsymbol{x}^*_0$ of (5.27) refers to the initial state at the current time instant, whereas $\boldsymbol{x}^*_0$ of (5.4) refers to the initial state at the start of the fundamental period.

Compact Vector Form of the Small-Signal Error

The small-signal error of (5.29) can be written in a compact vector form. First, the small-signal input $\tilde{\boldsymbol{u}}_{abc}$ is decomposed into each phase, resulting in (5.29) becoming

$$\tilde{\boldsymbol{x}}(t,\lambda_{p,i}) = \boldsymbol{e}^{\boldsymbol{F}t}\tilde{\boldsymbol{x}}_0 + \boldsymbol{\Gamma}(t) + \int_0^t \boldsymbol{e}^{\boldsymbol{F}(t-\tau)} \left(\boldsymbol{G}_a \sum_{i=1}^{n_a} \lambda_{a,i}\delta(\tau - t'_{a,i}) + \boldsymbol{G}_b \sum_{i=1}^{n_b} \lambda_{b,i}\delta(\tau - t'_{b,i}) + \boldsymbol{G}_c \sum_{i=1}^{n_c} \lambda_{c,i}\delta(\tau - t'_{c,i}) \right) \mathrm{d}\tau,$$

where $\boldsymbol{G}_a = \boldsymbol{G}[1\ 0\ 0]^{\mathrm{T}}$, $\boldsymbol{G}_b = \boldsymbol{G}[0\ 1\ 0]^{\mathrm{T}}$, and $\boldsymbol{G}_c = \boldsymbol{G}[0\ 0\ 1]^{\mathrm{T}}$; and

$$\boldsymbol{\Gamma}(t) = \int_0^t \boldsymbol{e}^{\boldsymbol{F}(t-\tau)}\boldsymbol{G}\bar{\boldsymbol{u}}_{abc}(\tau)\ \mathrm{d}\tau. \tag{5.30}$$

Derivations involving $\boldsymbol{\Gamma} \in \mathbb{R}^{n_x}$ are moved to Appendix D.1 for brevity. By using the well-known sifting property of the impulse function (see Appendix C), the small-signal error becomes

$$\tilde{\boldsymbol{x}}(t,\lambda_{p,i}) = \boldsymbol{e}^{\boldsymbol{F}t}\tilde{\boldsymbol{x}}_0 + \boldsymbol{\Gamma}(t) + \left(\sum_{i=1}^{n_a} \boldsymbol{e}^{\boldsymbol{F}(t-t'_{a,i})}\boldsymbol{G}_a h(t-t'_{a,i})\lambda_{a,i} + \sum_{i=1}^{n_b} \boldsymbol{e}^{\boldsymbol{F}(t-t'_{b,i})}\boldsymbol{G}_b h(t-t'_{b,i})\lambda_{b,i} + \sum_{i=1}^{n_c} \boldsymbol{e}^{\boldsymbol{F}(t-t'_{c,i})}\boldsymbol{G}_c h(t-t'_{c,i})\lambda_{c,i} \right).$$

By using basic algebraic manipulations, the small-signal error can be compactly stated as

$$\tilde{\boldsymbol{x}}(t,\boldsymbol{\Lambda}) = \boldsymbol{e}^{\boldsymbol{F}t}\tilde{\boldsymbol{x}}_0 + \boldsymbol{\Gamma}(t) + \boldsymbol{\Phi}(t)\boldsymbol{\Lambda} \tag{5.31}$$

where $\boldsymbol{\Phi} \in \mathbb{R}^{n_x \times n_{\mathrm{sw}}}$ is the input matrix,

$$\boldsymbol{\Phi}(t) = \begin{bmatrix} \boldsymbol{e}^{\boldsymbol{F}(t-t'_{a,1})}\boldsymbol{G}_a h(t-t'_{a,1}) & \cdots & \boldsymbol{e}^{\boldsymbol{F}(t-t'_{a,n_a})}\boldsymbol{G}_a h(t-t'_{a,n_a}) \\ \boldsymbol{e}^{\boldsymbol{F}(t-t'_{b,1})}\boldsymbol{G}_b h(t-t'_{b,1}) & \cdots & \boldsymbol{e}^{\boldsymbol{F}(t-t'_{b,n_b})}\boldsymbol{G}_b h(t-t'_{b,n_b}) \\ \boldsymbol{e}^{\boldsymbol{F}(t-t'_{c,1})}\boldsymbol{G}_c h(t-t'_{c,1}) & \cdots & \boldsymbol{e}^{\boldsymbol{F}(t-t'_{c,n_c})}\boldsymbol{G}_c h(t-t'_{c,n_c}) \end{bmatrix}, \tag{5.32}$$

and $\boldsymbol{\Lambda} \in \mathbb{R}^{n_{\mathrm{sw}}}$ is introduced as the *strength vector*,

$$\boldsymbol{\Lambda} = \begin{bmatrix} \lambda_{a,1} & \cdots & \lambda_{a,n_a} & \lambda_{b,1} & \cdots & \lambda_{b,n_b} & \lambda_{c,1} & \cdots & \lambda_{c,n_c} \end{bmatrix}^{\mathrm{T}}, \tag{5.33}$$

with the n_{sw} impulse strengths over the prediction horizon T_p. As an example, the strength vector corresponding to Figure 5.6 is

$$\boldsymbol{\Lambda} = \begin{bmatrix} \lambda_{a,1} & \lambda_{a,2} & \lambda_{b,1} & \lambda_{b,2} & \lambda_{b,3} & \lambda_{b,4} & \lambda_{c,1} & \lambda_{c,2} & \lambda_{c,3} \end{bmatrix}^{\mathrm{T}}. \tag{5.34}$$

Observe that the small-signal error is linear in the impulse strengths; if the time modifications are not represented by impulse strengths, and instead is an argument in the step function [as is the case with the nonlinearized pulse pattern of (5.13)], the time modifications will be the arguments of matrix exponentials after integrating the differential equations.

5.5.2 Objective Function

The control objectives are to minimize the small-signal error (which relates to the tracking error) across the prediction horizon and to penalize modifications to the incumbent pulse pattern (which relates to the control effort).[8] These control objectives can be mapped into a quadratic objective function as[9]

$$J(\boldsymbol{\Lambda}) = \int_0^{T_p} \tfrac{1}{2}\|\tilde{\boldsymbol{x}}(t,\boldsymbol{\Lambda})\|_{\boldsymbol{Q}}^2 \,\mathrm{d}t + \tfrac{1}{2}\|\boldsymbol{\Lambda}\|_{\boldsymbol{R}}^2. \tag{5.35}$$

The first term penalizes with the positive semidefinite diagonal (penalty) matrix $\boldsymbol{Q} \in \mathbb{R}^{n_x \times n_x}$ the integral of the small-signal error over the prediction horizon T_p. The diagonal (penalty) matrix $\boldsymbol{R} \in \mathbb{R}^{n_{\text{sw}} \times n_{\text{sw}}}$ corresponding to the penalty on the switching modifications is also required to be positive semidefinite. The penalty on the control effort inhibits the controller being overly aggressive. This is useful when modelling errors (recall that modifications to a pulse pattern are approximated and not exact) and noise (from measurement devices and observers) are present in a system. The objective function can now be expanded and written in standard quadratic form.

Tracking Error Term

Consider the tracking error term[10]

$$J_1(\boldsymbol{\Lambda}) = \int_0^{T_p} \tfrac{1}{2}\|\tilde{\boldsymbol{x}}(t,\boldsymbol{\Lambda})\|_{\boldsymbol{Q}}^2 \,\mathrm{d}t,$$

which, after inserting (5.31), is expanded to

$$\begin{aligned} J_1(\boldsymbol{\Lambda}) &= \int_0^{T_p} \tfrac{1}{2}\left(\boldsymbol{e}^{\boldsymbol{F}t}\tilde{\boldsymbol{x}}_0 + \boldsymbol{\Gamma}(t) + \boldsymbol{\Phi}(t)\boldsymbol{\Lambda}\right)^{\mathrm{T}} \boldsymbol{Q}\left(\boldsymbol{e}^{\boldsymbol{F}t}\tilde{\boldsymbol{x}}_0 + \boldsymbol{\Gamma}(t) + \boldsymbol{\Phi}(t)\boldsymbol{\Lambda}\right)\,\mathrm{d}t \\ &= \tfrac{1}{2}\boldsymbol{\Lambda}^{\mathrm{T}}\int_0^{T_p} \boldsymbol{\Upsilon}(t)\,\mathrm{d}t\;\boldsymbol{\Lambda} + \int_0^{T_p} \boldsymbol{\Theta}^{\mathrm{T}}(t)\,\mathrm{d}t\;\boldsymbol{\Lambda} + \int_0^{T_p} \theta(t)\,\mathrm{d}t, \end{aligned} \tag{5.36}$$

where $\boldsymbol{\Upsilon} \in \mathbb{R}^{n_{\text{sw}} \times n_{\text{sw}}}$, $\boldsymbol{\Theta} \in \mathbb{R}^{n_{\text{sw}}}$, and $\theta \in \mathbb{R}$ are defined as

$$\boldsymbol{\Upsilon}(t) = \boldsymbol{\Phi}^{\mathrm{T}}(t)\boldsymbol{Q}\boldsymbol{\Phi}(t) \tag{5.37}$$

$$\boldsymbol{\Theta}^{\mathrm{T}}(t) = \left(\boldsymbol{e}^{\boldsymbol{F}t}\tilde{\boldsymbol{x}}_0 + \boldsymbol{\Gamma}(t)\right)^{\mathrm{T}}\boldsymbol{Q}\boldsymbol{\Phi}(t) \tag{5.38}$$

$$\theta(t) = \tfrac{1}{2}\left(\boldsymbol{e}^{\boldsymbol{F}t}\tilde{\boldsymbol{x}}_0 + \boldsymbol{\Gamma}(t)\right)^{\mathrm{T}}\boldsymbol{Q}\left(\boldsymbol{e}^{\boldsymbol{F}t}\tilde{\boldsymbol{x}}_0 + \boldsymbol{\Gamma}(t)\right), \tag{5.39}$$

respectively.

First, consider the (i',j')th entry of $\boldsymbol{\Upsilon}$ that is to be integrated. The i'th entry corresponds to phase $p_1 \in \{a,b,c\}$ and the ith switching transition in that phase. With (5.34) as an example, $i' = 5$ refers to phase $p_1 = b$ and to the switching transition $i = 3$. The j'th entry is defined accordingly with phase $p_2 \in \{a,b,c\}$ and the jth switching transition in that phase. From (5.37), and by inserting (5.32), the (i',j')th entry of $\boldsymbol{\Upsilon}$ follows as

$$\begin{aligned} \boldsymbol{\Upsilon}_{(i',j')}(t) &= \left(\boldsymbol{e}^{\boldsymbol{F}(t-t'_{p_1,i})}\boldsymbol{G}_{p_1}h(t-t'_{p_1,i})\right)^{\mathrm{T}}\boldsymbol{Q}\boldsymbol{e}^{\boldsymbol{F}(t-t'_{p_2,j})}\boldsymbol{G}_{p_2}h(t-t'_{p_2,j}) \\ &= \boldsymbol{G}_{p_1}^{\mathrm{T}}\boldsymbol{e}^{\boldsymbol{F}^{\mathrm{T}}(t-t'_{p_1,i})}\boldsymbol{Q}\boldsymbol{e}^{\boldsymbol{F}(t-t'_{p_2,j})}\boldsymbol{G}_{p_2}h(t-t'_{p,ij}), \end{aligned}$$

[8] Alternatively, the control effort term can instead penalize the deviation between the modified pulse pattern and the nominal pulse pattern.

[9] The scaling factor of $\frac{1}{2}$ is added to ensure the problem can be written in standard quadratic form.

[10] Note that $\|\boldsymbol{\xi}\|_{\boldsymbol{Q}}^2 = \boldsymbol{\xi}^{\mathrm{T}}\boldsymbol{Q}\boldsymbol{\xi}$, the 2-norm of vector $\boldsymbol{\xi}$ with matrix $\boldsymbol{Q}$.

where $t'_{p,ij} = \max\{t'_{p_1,i}, t'_{p_2,j}\}$. Note that its integral

$$\boldsymbol{V}_{(i',j')} = \int_0^{T_p} \boldsymbol{G}_{p_1}^{\mathrm{T}} e^{\boldsymbol{F}^{\mathrm{T}}(t-t'_{p_1,i})} \boldsymbol{Q} e^{\boldsymbol{F}(t-t'_{p_2,j})} \boldsymbol{G}_{p_2} h(t - t'_{p,ij}) \, \mathrm{d}t \tag{5.40}$$

involves a product of two matrix exponentials. In general, these matrices do not commute and thus the integral is not trivial to solve. Fortunately, an extremely useful theorem from [58] states that the integral

$$\boldsymbol{\Xi}(\Delta T) = \int_0^{\Delta T} e^{\boldsymbol{F}^{\mathrm{T}} t} \boldsymbol{Q} e^{\boldsymbol{F} t} \, \mathrm{d}t \tag{5.41}$$

can be solved with

$$\boldsymbol{\Xi}(\Delta T) = \boldsymbol{M}^{\mathrm{T}}(\Delta T) \boldsymbol{N}(\Delta T), \tag{5.42}$$

where

$$\begin{bmatrix} \boldsymbol{L}(\Delta T) & \boldsymbol{N}(\Delta T) \\ \boldsymbol{0}_{n_x \times n_x} & \boldsymbol{M}(\Delta T) \end{bmatrix} = e^{\begin{bmatrix} -\boldsymbol{F}^{\mathrm{T}} & \boldsymbol{Q} \\ \boldsymbol{0}_{n_x \times n_x} & \boldsymbol{F} \end{bmatrix} \Delta T} = \begin{bmatrix} e^{-\boldsymbol{F}^{\mathrm{T}} \Delta T} & e^{-\boldsymbol{F}^{\mathrm{T}} \Delta T} \int_0^{\Delta T} e^{\boldsymbol{F}^{\mathrm{T}} t} \boldsymbol{Q} e^{\boldsymbol{F} t} \, \mathrm{d}t \\ \boldsymbol{0}_{n_x \times n_x} & e^{\boldsymbol{F} \Delta T} \end{bmatrix}.$$

To use this theorem, the integral of (5.40) needs to be manipulated into the form of (5.41) so that (5.42) can be used. By knowing that the step function changes the lower bound of an integral, (5.40) becomes

$$\boldsymbol{V}_{(i',j')} = \boldsymbol{G}_{p_1}^{\mathrm{T}} \int_{t'_{p,ij}}^{T_p} e^{\boldsymbol{F}^{\mathrm{T}}(t-t'_{p_1,i})} \boldsymbol{Q} e^{\boldsymbol{F}(t-t'_{p_2,j})} \, \mathrm{d}t \, \boldsymbol{G}_{p_2},$$

and by shifting the integrand forward by $t'_{p,ij}$ (and using some basic manipulations),

$$\boldsymbol{V}_{(i',j')} = \boldsymbol{G}_{p_1}^{\mathrm{T}} e^{\boldsymbol{F}^{\mathrm{T}}(t'_{p,ij}-t'_{p_1,i})} \int_0^{T_p - t'_{p,ij}} e^{\boldsymbol{F}^{\mathrm{T}} t} \boldsymbol{Q} e^{\boldsymbol{F} t} \, \mathrm{d}t \, e^{\boldsymbol{F}(t'_{p,ij}-t'_{p_2,j})} \boldsymbol{G}_{p_2},$$

which is then stated as

$$\boldsymbol{V}_{(i',j')} = \boldsymbol{G}_{p_1}^{\mathrm{T}} e^{\boldsymbol{F}^{\mathrm{T}}(t'_{p,ij}-t'_{p_1,i})} \boldsymbol{\Xi}(T_p - t'_{p,ij}) e^{\boldsymbol{F}(t'_{p,ij}-t'_{p_2,j})} \boldsymbol{G}_{p_2} \tag{5.43}$$

using the definition of (5.41).

Next consider $\boldsymbol{\Theta}$ of (5.38), which is split into two terms

$$\boldsymbol{\Theta}^{\mathrm{T}}(t) = \boldsymbol{\Theta}_0^{\mathrm{T}}(t) + \boldsymbol{\Theta}_\Gamma^{\mathrm{T}}(t)$$

where

$$\begin{aligned} \boldsymbol{\Theta}_0^{\mathrm{T}}(t) &= \left(e^{\boldsymbol{F} t} \tilde{\boldsymbol{x}}_0\right)^{\mathrm{T}} \boldsymbol{Q} \boldsymbol{\Phi}(t) \\ \boldsymbol{\Theta}_\Gamma^{\mathrm{T}}(t) &= \boldsymbol{\Gamma}^{\mathrm{T}}(t) \boldsymbol{Q} \boldsymbol{\Phi}(t). \end{aligned}$$

Consider the i'th entry of $\boldsymbol{\Theta}_0^{\mathrm{T}}$ (which refers to phase p and the ith switching transition therein), which, after inserting (5.32), follows as

$$\boldsymbol{\Theta}_{0,i'}^{\mathrm{T}}(t) = \tilde{\boldsymbol{x}}_0^{\mathrm{T}} e^{\boldsymbol{F}^{\mathrm{T}} t} \boldsymbol{Q} e^{\boldsymbol{F}(t-t'_{p,i})} \boldsymbol{G}_p h(t - t'_{p,i}).$$

Its integral

$$\boldsymbol{c}_{0,i'}^{\mathrm{T}} = \int_0^{T_p} \tilde{\boldsymbol{x}}_0^{\mathrm{T}} e^{\boldsymbol{F}^{\mathrm{T}} t} \boldsymbol{Q} e^{\boldsymbol{F}(t-t'_{p,i})} \boldsymbol{G}_p h(t - t'_{p,i}) \, \mathrm{d}t \tag{5.44}$$

can also be written in the form of (5.41). By following similar manipulations used in the derivation of (5.43), the integral of (5.44) becomes

$$\boldsymbol{c}_{0,i'}^{\mathrm{T}} = \tilde{\boldsymbol{x}}_0^{\mathrm{T}} \boldsymbol{e}^{\boldsymbol{F}^{\mathrm{T}} t'_{p,i}} \int_0^{T_p - t'_{p,i}} \boldsymbol{e}^{\boldsymbol{F}^{\mathrm{T}} t} \boldsymbol{Q} \boldsymbol{e}^{\boldsymbol{F} t} \, \mathrm{d}t \, \boldsymbol{G}_p,$$

and is stated as

$$\boldsymbol{c}_{0,i'}^{\mathrm{T}} = \tilde{\boldsymbol{x}}_0^{\mathrm{T}} \boldsymbol{e}^{\boldsymbol{F}^{\mathrm{T}} t'_{p,i}} \boldsymbol{\Xi}(T_p - t'_{p,i}) \boldsymbol{G}_p \tag{5.45}$$

according to (5.41). The manipulations of the integral of the second term,

$$\boldsymbol{c}_{\Gamma}^{\mathrm{T}} = \int_0^{T_p} \boldsymbol{\Theta}_{\Gamma}^{\mathrm{T}}(t) \, \mathrm{d}t, \tag{5.46}$$

are moved to Appendix D.2 for brevity, since they involve similar steps used in the derivation of (5.43). After $\boldsymbol{c}_0$ and $\boldsymbol{c}_{\Gamma}$ are calculated, $\boldsymbol{c}$ follows as

$$\boldsymbol{c} = \boldsymbol{c}_0 + \boldsymbol{c}_{\Gamma}. \tag{5.47}$$

The tracking error term can now be compactly stated as

$$J_1(\boldsymbol{\Lambda}) = \tfrac{1}{2} \boldsymbol{\Lambda}^{\mathrm{T}} \boldsymbol{V} \boldsymbol{\Lambda} + \boldsymbol{c}^{\mathrm{T}} \boldsymbol{\Lambda}. \tag{5.48}$$

Note the $\theta(t)$ of (5.39) is a constant term in the objective function and is therefore omitted, since it has no influence on the minimum.

Control Effort Term

The control effort term of (5.35) is simply

$$J_2(\boldsymbol{\Lambda}) = \tfrac{1}{2} \boldsymbol{\Lambda}^{\mathrm{T}} \boldsymbol{R} \boldsymbol{\Lambda}. \tag{5.49}$$

Quadratic Form

Using the results of (5.48) and (5.49), the objective function can now be written in its standard quadratic form as

$$\begin{aligned} J(\boldsymbol{\Lambda}) &= J_1(\boldsymbol{\Lambda}) + J_2(\boldsymbol{\Lambda}) \\ &= \tfrac{1}{2} \boldsymbol{\Lambda}^{\mathrm{T}} \boldsymbol{H} \boldsymbol{\Lambda} + \boldsymbol{c}^{\mathrm{T}} \boldsymbol{\Lambda}, \end{aligned} \tag{5.50}$$

where $\boldsymbol{H} = \boldsymbol{V} + \boldsymbol{R}$ is the Hessian and $\boldsymbol{c}$ the vector with linear coefficients.

5.5.3 Constraints

During minimization of (5.50), constraints are required on the switching time modifications to ensure a feasible pulse pattern. Similar to the linear constraints imposed when calculating an OPP [see (3.25)], the $n_p + 1$ constraints

$$0 \leq t_{p,1} \leq t_{p,2} \leq \cdots \leq t_{p,n_p} \leq T_p \tag{5.51}$$

are imposed for each phase $p \in \{a, b, c\}$. These constraints ensure that the switching transitions are in ascending order, nonnegative, and are not moved beyond the prediction horizon. Alternatively, the n_pth switching transition can be upper bounded by the first switching transition that occurs

after the prediction horizon, t_{p,n_p+1}. Since the impulse strengths $\lambda_{p,i}$ are the decision variables, the modified switching instants $t_{p,i}$ are written in terms of the impulse strengths as

$$0 \leq t'_{p,1} - \frac{\lambda_{p,1}}{\Delta u_{p,1}} \leq t'_{p,2} - \frac{\lambda_{p,2}}{\Delta u_{p,2}} \leq \cdots \leq t'_{p,n_p} - \frac{\lambda_{p,n_p}}{\Delta u_{p,n_p}} \leq T_p, \tag{5.52}$$

where the definitions of (5.19) and (5.20) are used.

By applying (5.52) to all three phases, the constraints in matrix notation follow as

$$\boldsymbol{A}\boldsymbol{\Lambda} \leq \boldsymbol{b} \tag{5.53}$$

where $\boldsymbol{A} \in \mathbb{R}^{(n_{sw}+3)\times n_{sw}}$ and $\boldsymbol{b} \in \mathbb{R}^{n_{sw}+3}$ are defined as

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{A}_a & & \\ & \boldsymbol{A}_b & \\ & & \boldsymbol{A}_c \end{bmatrix} \text{ and } \boldsymbol{b} = \begin{bmatrix} \boldsymbol{b}_a \\ \boldsymbol{b}_b \\ \boldsymbol{b}_c \end{bmatrix},$$

respectively, with the per-phase constraint matrix $\boldsymbol{A}_p \in \mathbb{R}^{(n_p+1)\times n_p}$ (which is an augmented first-order difference matrix) and vector $\boldsymbol{b}_p \in \mathbb{R}^{n_p+1}$ following as

$$\boldsymbol{A}_p = \begin{bmatrix} \frac{1}{\Delta u_{p,1}} & 0 & 0 & \cdots & 0 & 0 \\ -\frac{1}{\Delta u_{p,1}} & \frac{1}{\Delta u_{p,2}} & 0 & \cdots & 0 & 0 \\ 0 & -\frac{1}{\Delta u_{p,2}} & \frac{1}{\Delta u_{p,3}} & & 0 & 0 \\ \vdots & & & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & -\frac{1}{\Delta u_{p,n_p-1}} & \frac{1}{\Delta u_{p,n_p}} \\ 0 & 0 & 0 & \cdots & 0 & \frac{1}{\Delta u_{p,n_p}} \end{bmatrix}$$

and

$$\boldsymbol{b}_p = \begin{bmatrix} t'_{p,1} & t'_{p,2} - t'_{p,1} & \cdots & T_p - t'_{p,n_p} \end{bmatrix}^{\mathrm{T}},$$

respectively.

5.5.4 Optimization

Minimizing (5.50) subject to the constraints of (5.53) results in the QP

$$\boldsymbol{\Lambda}_{\text{opt}} = \arg\min_{\boldsymbol{\Lambda}} \quad \tfrac{1}{2}\boldsymbol{\Lambda}^{\mathrm{T}}\boldsymbol{H}\boldsymbol{\Lambda} + \boldsymbol{c}^{\mathrm{T}}\boldsymbol{\Lambda} \tag{5.54a}$$

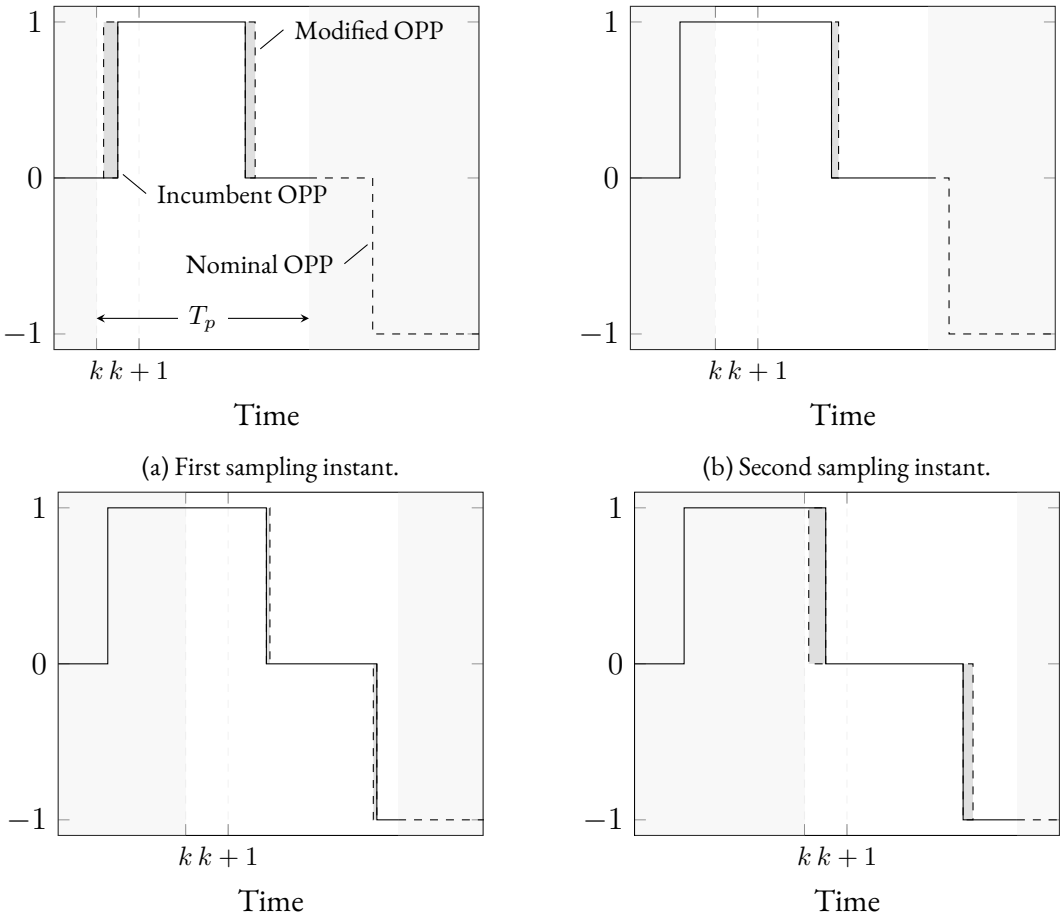
$$\text{subject to } \boldsymbol{A}\boldsymbol{\Lambda} \leq \boldsymbol{b}, \tag{5.54b}$$

where $\boldsymbol{\Lambda}_{\text{opt}}$ is the (open-loop) optimal strength vector. The Hessian $\boldsymbol{H}$ is guaranteed to be at least positive semidefinite and therefore the problem is convex (see Appendix E for an analysis on the definiteness of the Hessian). In Chapter 8, the QP of (5.54) is solved using the gradient projection method of Section 2.3.2, whereas in all other chapters CPLEX (which is an optimization suite developed by IBM) is used.

Optimal Switching Instants

Once the optimal strength vector $\boldsymbol{\Lambda}_{\text{opt}}$ is calculated by solving the QP of (5.54), the impulse strengths are translated into switching time modifications [by using (5.20)] as

$$\Delta t_{\text{opt},p,i} = -\frac{\lambda_{\text{opt},p,i}}{\Delta u_{p,i}},$$



(a) First sampling instant.

(b) Second sampling instant.

(c) Third sampling instant.

(d) Fourth sampling instant.

Figure 5.7: The receding horizon policy.

which are then added to the incumbent switching instants [according to (5.19)],

$$t_{\text{opt},p,i} = t'_{p,i} + \Delta t_{\text{opt},p,i},$$

to yield the optimal modified switching instants.

5.5.5 Receding Horizon

The control algorithm operates at discrete-time instants of kT_s, where $k \in \mathbb{N}$ is the discrete instant and T_s the sampling interval. Note that although the control algorithm operates at discrete instants, the control problem is formulated in the continuous-time domain.

Out of the long prediction interval T_p, only the modified switching instants within the current sampling interval [that is, between kT_s and $(k+1)T_s$] are applied to the converter system. At each subsequent sampling instant k, new measurements are taken and the incumbent switching instants (which are the modified switching instants calculated at the previous sampling instant $k-1$) are re-optimized based on new information. Since the modified switching instants are calculated in an open-loop manner, this process, known as the *receding horizon* policy, is key to providing feedback and making the controller more robust to measurement and modelling errors. Figure 5.7 illustrates the receding horizon policy. In Figures 5.7a–5.7c, no measurement errors or sudden changes in operating conditions are present; the controller makes small adjustments to the modified switching instants at each subsequent sampling instant. However, in Figure 5.7d, there is a sudden change

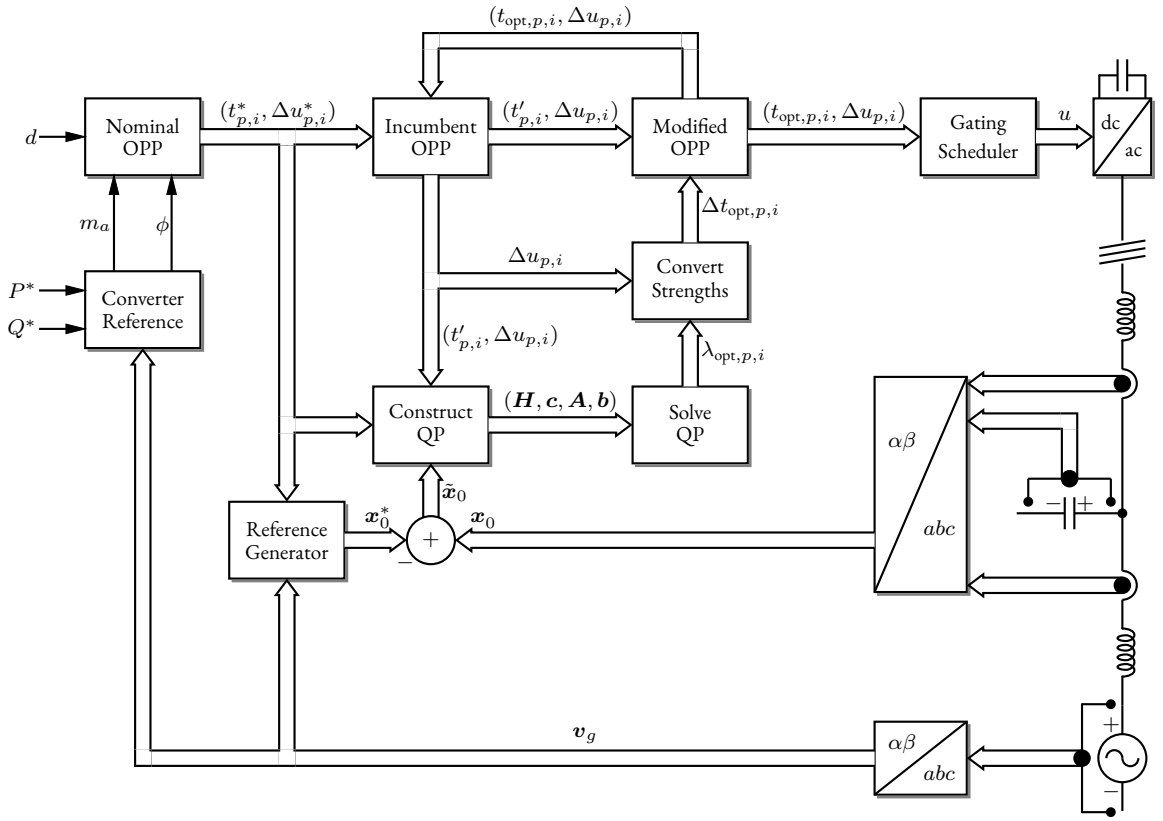


Figure 5.8: Block diagram of small-signal controller.

in the operating conditions (such as a reference step or fault). The controller then significantly modifies the pulse pattern based on the new information. This highlights the importance of the receding horizon policy and demonstrates why the controller actions determined across the (long) prediction horizon should not be applied in open-loop fashion.

5.5.6 Control Algorithm

The control diagram of the small-signal controller is shown in Figure 5.8. The overall control algorithm consists of the following steps:

1. The real power P^* and reactive power Q^* references are mapped into a modulation index m_a and pulse pattern phase ϕ, which depends on the grid voltage $\boldsymbol{v}_g$, in the *Converter Reference* block via simple phasor analysis.

2. For the given modulation index m_a and the selected pulse number d, the nominal pulse pattern is read out from a lookup table and phase-shifted accordingly by ϕ in the *Nominal OPP* block.

3. The *Reference Generator* block calculates the steady-state trajectory $\boldsymbol{x}^*$ over one fundamental period according to the method of Section 5.3. Alternatively, if storage is not limited, all steady-state trajectories corresponding to their nominal pulse patterns can be calculated offline and stored in lookup tables.

4. The initial small-signal error is calculated as $\tilde{\boldsymbol{x}}_0 = \boldsymbol{x}_0 - \boldsymbol{x}^*_0$, where the steady-state value at the current instant $\boldsymbol{x}^*_0$ is read-out from the *Reference Generator* block and $\boldsymbol{x}_0$ is the measured state vector.

5. Given the small-signal error, and the nominal and incumbent switching transitions that fall within the prediction horizon, the *Construct QP* block calculates the Hessian $\boldsymbol{H}$ and vector $\boldsymbol{c}$, as well as the constraint matrix $\boldsymbol{A}$ and vector $\boldsymbol{b}$.

6. Once the *Solve QP* block solves (5.54), the optimal impulse strengths $\lambda_{\text{opt},p,i}$ are converted into optimal switching time modifications $\Delta t_{\text{opt},p,i}$ in the *Convert Strengths* block.

7. After the (optimal) modified switching instants are calculated in the *Modified OPP* block as $t_{\text{opt},p,i} = t'_{p,i} + \Delta t_{\text{opt},p,i}$, the *Gating Scheduler* block determines the switching transitions that fall within the current sampling interval (which are also then removed from the modified pulse pattern).

8. At the subsequent sampling instant, the incumbent pulse pattern is updated with the modified pulse pattern in the *Incumbent OPP* block.

Note that Steps 1 to 3 are only required when the operating conditions change.

5.5.7 Standard Control Algorithm

Note that the updating of the incumbent pulse pattern (that is, the successive re-linearization of the pulse pattern at the newly-calculated switching instants) is an optional step, with the aim of improving the accuracy of the state predictions. By neglecting this step, the computational burden of the controller can be reduced. This implies that the nominal pulse pattern $\boldsymbol{u}^*_{abc}$ and incumbent pulse pattern $\boldsymbol{u}_{abc}$ are equal, which results in $\bar{\boldsymbol{u}}_{abc}$ being zero. From (5.31), this leads to the model of the small-signal error reducing to

$$\tilde{\boldsymbol{x}}(t, \boldsymbol{\Lambda}) = \boldsymbol{e}^{\boldsymbol{F}t}\tilde{\boldsymbol{x}}_0 + \boldsymbol{\Phi}(t)\boldsymbol{\Lambda}. \tag{5.55}$$

It can easily be verified that the calculations regarding the Hessian $\boldsymbol{H}$ are identical. However, the calculations regarding the vector with linear coefficients reduces to $\boldsymbol{c} = \boldsymbol{c}_0$ [see (5.47)], resulting in a moderate decrease in computational burden. Unfortunately, and as expected, the accuracy of the state predictions will decrease as the modifications increase. Nevertheless, these inaccuracies can be compensated for by the controller thanks to the receding horizon policy.

In the sequel, the small-signal controller that uses the reduced model of (5.55) is referred to as the *standard* controller, whereas if (5.31) is used it is referred to as the *advanced* controller.

5.6 Performance Evaluation

This section evaluates the efficacy of the small-signal controller via simulations. The performance of the advanced controller during multiple reference steps is first demonstrated. Thereafter, the response of the standard controller is compared to that of the advanced controller. Also, the accuracy of the predictions of the two controller variants is compared. Finally, the response of the advanced controller is compared to a controller with a nonlinear model (in other words, if impulses are not used to represent modifications) in order to establish an upper bound in the achievable performance of the small-signal controller. This controller is simply referred to as the *nonlinearized* controller.

As a case study, the grid-connected converter system from Section 3.3.2 is used. The device switching frequency is chosen as $f_{\text{sw}} = 250\,\text{Hz}$ (a pulse number of $d = 5$), which is a typical switching frequency of medium-voltage systems. The nominal pulse pattern is calculated so that the harmonic distortion of the grid current is minimized. Unless mentioned otherwise, the prediction horizon of the controller is set to $T_p = 2\,\text{ms}$; this is a practically reasonable horizon, usually resulting

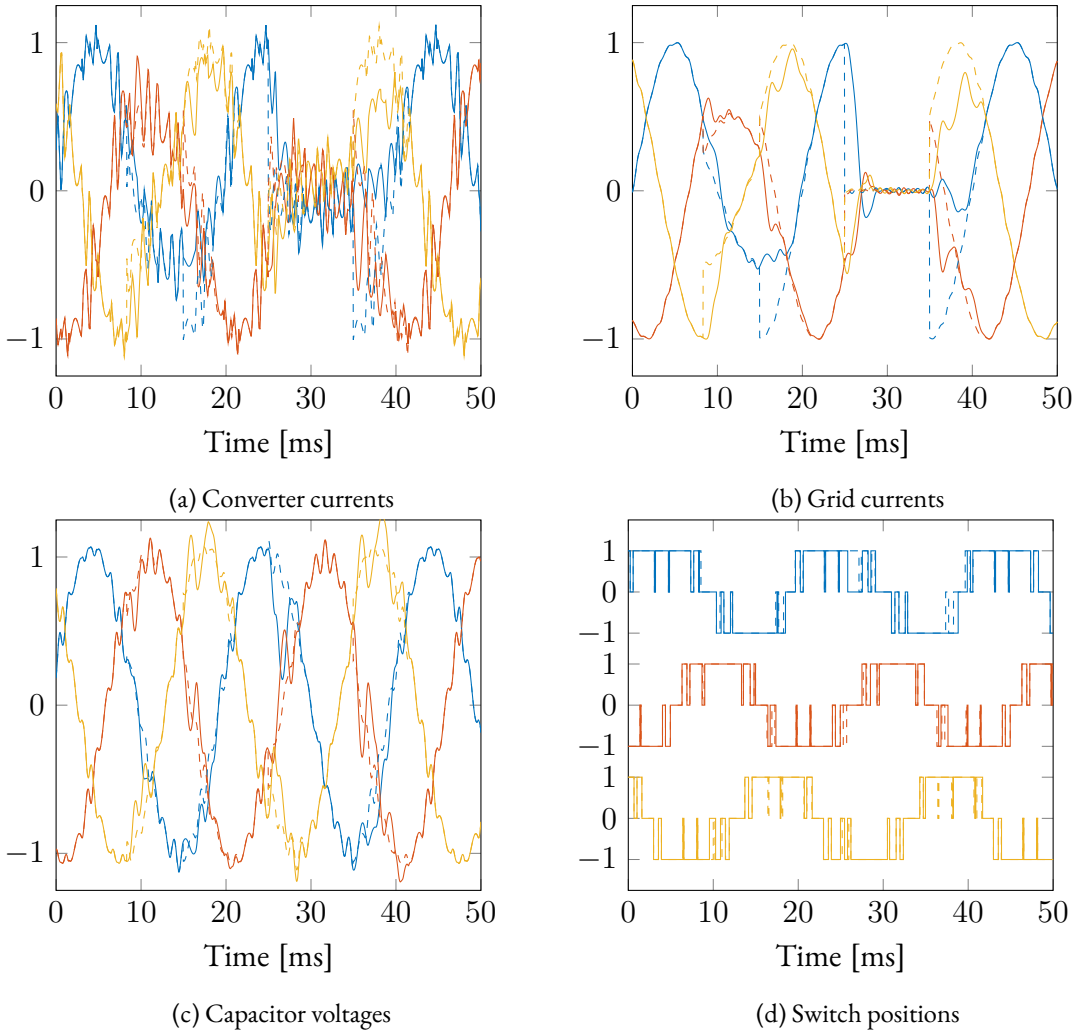


Figure 5.9: The response of the converter states (in pu) during multiple reference steps. References are indicated with dashed lines.

in two or three switching transitions per phase for the given pulse number of $d = 5$. All state variables are penalized equally with $\boldsymbol{Q} = \mathbf{I}_6$ and all switching transition modifications are also penalized equally with $\boldsymbol{R} = \mathbf{I}_{n_{\text{sw}}}$. The sampling interval of the controller is set to $T_s = 25\,\mu\text{s}$.

5.6.1 Response Time During Transients

The primary objective of the controller is to achieve a fast closed-loop response when a converter is modulated by OPPs. In Figure 5.9, the (advanced) controller is given four power reference steps. The reactive power reference Q^* is set to zero and thus the converter is operating at unity power factor. Initially, the converter is operating at rated power ($P = 1\,\text{pu}$), when the first reference step of $P^* = 0.5\,\text{pu}$ is applied at 8.35 ms. As soon as the controller has regulated the state vector onto its steady-state trajectory at 15 ms, the power reference is stepped back to $P^* = 1\,\text{pu}$. The power reference is then stepped to $P^* = 0\,\text{pu}$ at 25 ms and then stepped back again to $P^* = 1\,\text{pu}$ at 35 ms.

It is seen that the controller swiftly regulates all the state variables to their respective references. During the transients, it can be observed from Figure 5.9d that the nominal pulse pattern is modified significantly; for example, note that from 35 ms to 40 ms some of the switching transitions of phases a and c are removed. Once the converter enters the steady state, it can be seen that the (unmodified)

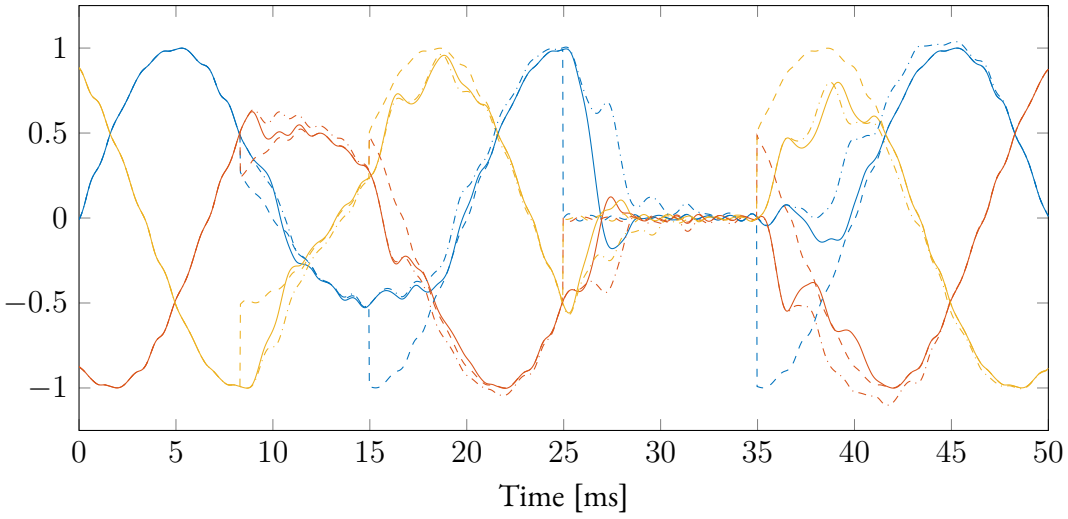


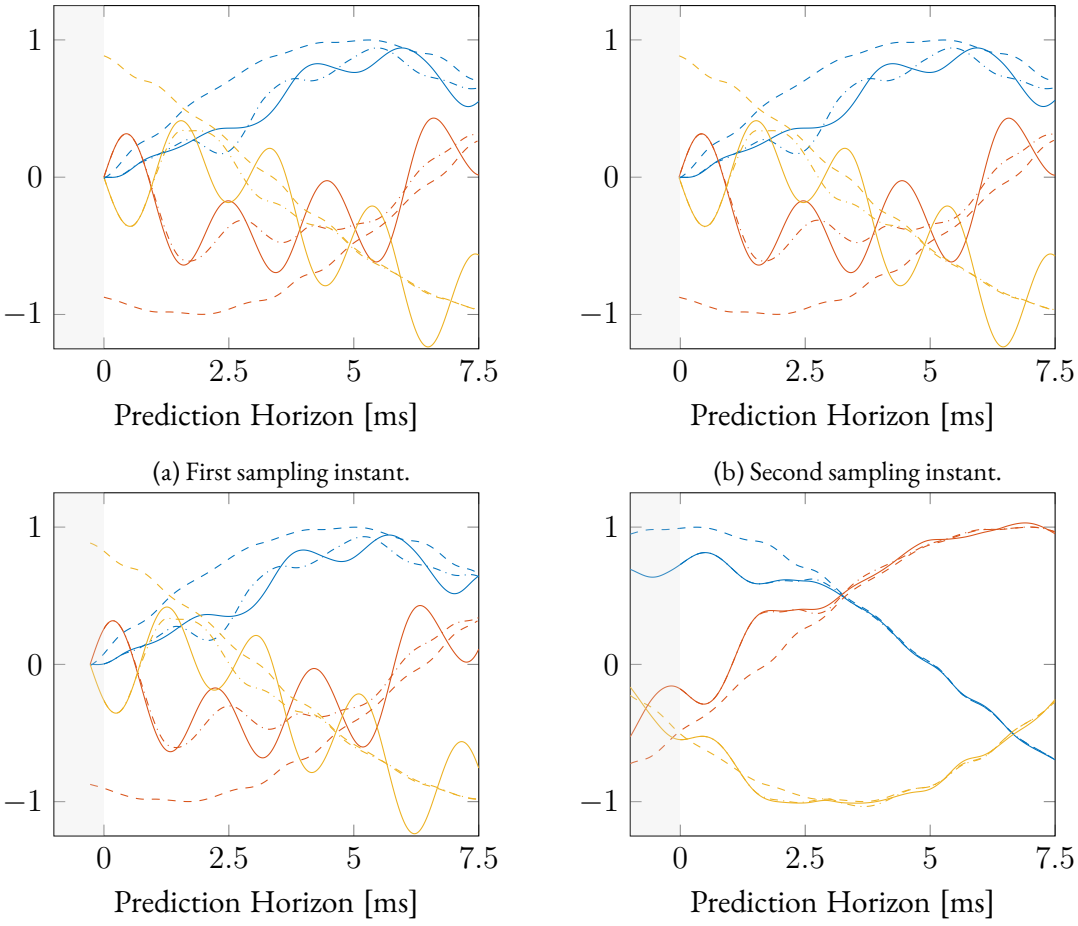
Figure 5.10: The grid current (in pu) responses of the standard and advanced controllers during reference steps. References are indicated by dashed lines. The responses of the standard and advanced controllers are indicated with dash-dotted and solid lines, respectively.

nominal pulse pattern is applied to the converter system; the superior harmonic performance of OPPs is achieved. Although there are some oscillations present during the transients, it should be noted that the converter system is underdamped and, due to the low switching frequency, there are only a few switching transitions available that can be used to achieve closed-loop control. It is likely that the response time can be reduced if additional switching transitions are inserted; this is known as *pulse insertion* and the MP^{3}C algorithm utilizes this technique, see [4, Section 12.6] and [59]. In the event that the oscillations are unacceptable, the reference steps can be limited by a ramp-limiter. However, this will increase the response time.

5.6.2 Standard Controller and Prediction Accuracy

Consider the standard controller described in Section 5.5.7, which is given the same reference steps as the advanced controller of the previous section. In Figure 5.10, the responses of the standard and advanced controllers are shown; only the grid currents are illustrated for clarity. As expected, the advanced controller noticeably outperforms its standard counterpart, although the standard controller does perform reasonably well.

For an illustration on how the superior internal dynamic model of the advanced controller enables it to outperform the standard controller, consider the state predictions of the standard and advanced controllers in Figures 5.11 and 5.12, respectively. For this illustration, the prediction horizon is increased to $T_p = 7.5$ ms, and all states are set to zero (the converter is at start-up). At the first sampling instant, it can be seen that the standard (see Figure 5.11a) and advanced (see Figure 5.12a) controllers have identical predictions; this is expected, since the incumbent pulse pattern of the advanced controller has not been updated. However, at the second sampling instant, the predictions of the advanced controller (see Figure 5.12b) significantly improve thanks to its ability to accurately predict the previous corrections; the incumbent pulse pattern has been updated. Note that the actual response has also changed. In contrast, the standard controller still has inaccurate predictions as shown in Figure 5.11b. At the 12th sampling instant, the advanced controller predicts the actual behaviour nearly perfectly as shown in Figure 5.12c; unless the operating conditions change, the advanced controller makes no further modifications to the incumbent pulse pattern. Regarding the standard model, it can be seen in Figure 5.11c that a margin of error is still present in its pre-



(a) First sampling instant.

(b) Second sampling instant.

(c) 12th sampling instant.

(d) 200th sampling instant.

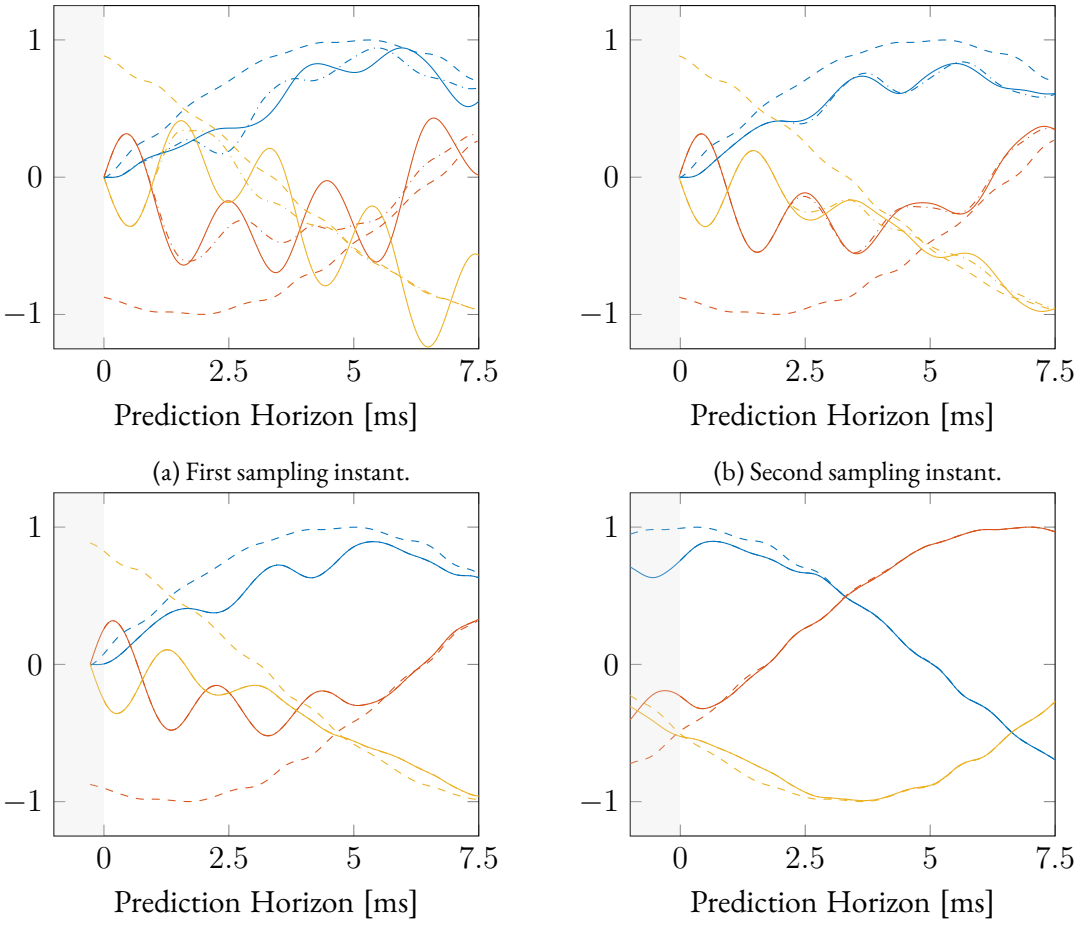
Figure 5.11: Grid current (in pu) prediction of the standard controller across a prediction horizon of $T_p = 7.5$ ms. The dashed lines are the references, the dash-dotted lines are the state predictions of the controller, and the solid lines are the actual responses if the pulse pattern is applied open loop.

dictions. Fortunately, thanks the receding horizon policy, the controller actions determined by the standard model are not applied open loop across the prediction horizon. As the system states reach their steady-state trajectories, it can be observed in Figure 5.11d that the predictions of the standard model are sufficiently accurate since only small modifications are necessary.

Note that the standard controller assumes that the nominal pulse pattern and incumbent pulse pattern are equal. This, however, is not always true. If switching transitions are delayed or advanced, there will be a mismatch between the nominal and incumbent pulse patterns that will manifest itself as an error; this is not compensated for. In contrast, the advanced controller does not suffer from this phenomenon and no compensation is required, since its internal model does not assume that the nominal and incumbent pulse patterns are equal.

5.6.3 Comparison to Nonlinearized Controller

In order to establish an upper limit to what is achievable with OPP-based model predictive controllers that have a similar formulation to the small-signal controller, its response is compared to that of a nonlinearized controller. Specifically, the nonlinearized controller does not use impulses to model corrections, and the time modifications are arguments of matrix exponentials. For the nonlinearized controller, the underlying optimization problem contains matrix exponentials that have



(a) First sampling instant.

(b) Second sampling instant.

(c) 12th sampling instant.

(d) 200th sampling instant.

Figure 5.12: Grid current (in pu) prediction of the advanced controller across a prediction horizon of $T_p = 7.5$ ms. The dashed lines are the references, the dash-dotted lines are the state predictions of the controller, and the solid lines are the actual responses if the pulse pattern is applied open loop.

to be evaluated at every iteration of the optimization algorithm, and therefore its computational burden is immense and the method is not practically viable. Moreover, the problem is nonconvex. The optimization problem of the nonlinearized controller is solved using the `fmincon` function of Matlab. Note that only a single initial solution (of zero modifications) is used.

In Figure 5.13, the responses of the grid currents for the (advanced) small-signal controller and the nonlinearized controller are shown. Interestingly, at most instances, the small-signal controller matches the response of the nonlinearized controller. This implies that the successive linearization step (around the previously-calculated modified switching instances) of the advanced small-signal controller is, in a sense, similar to sequential quadratic programming: the nonlinear objective function (with matrix exponentials) is approximated as a quadratic function at each sampling instant, and a QP is then solved to determine new modified switching instants. However, at certain instances, such as at 27 ms, the nonlinearized controller does have a quicker response. It should be noted that an even quicker response is possible by the nonlinearized controller if multiple initial conditions are used in order to find the global minimum (however, the already-high computational burden would increase even more). Nonetheless, the small-signal controller, with a relatively low computational burden, performs very well when compared to the nonlinearized controller.

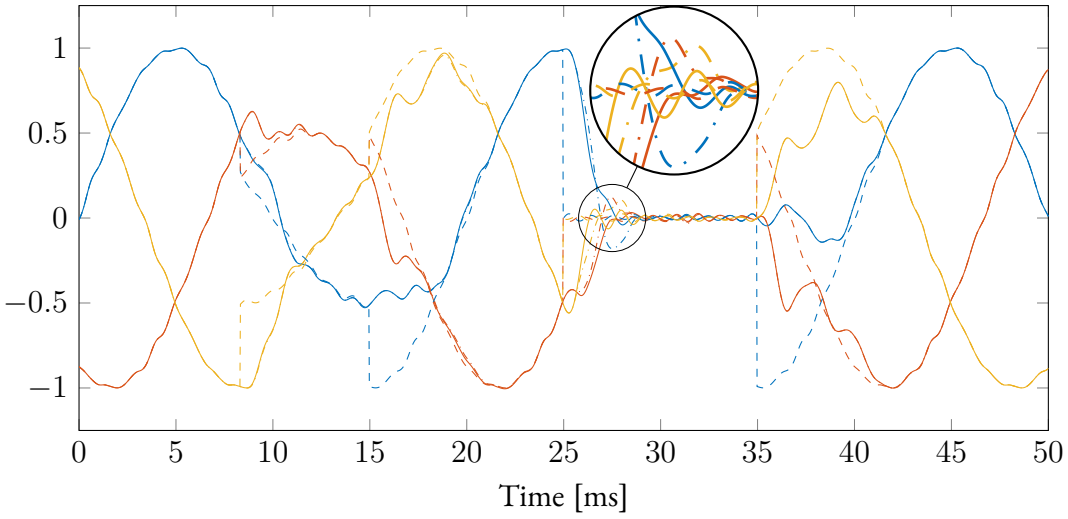


Figure 5.13: The grid current (in pu) responses during reference steps of the nonlinearized and advanced controllers. References are indicated by dashed lines. The responses of the nonlinearized and advanced controllers are indicated with solid and dash-dotted lines, respectively.

5.7 Summary

A generalized model predictive pulse pattern controller, which is referred to as the small-signal controller, was introduced. Thanks to the use of the strength of impulses to model corrections to a pulse pattern, the state vector is linear in the corrections. This resulted in the underlying optimization problem to be a QP. Although the use of impulses is an approximation and introduces an error in the state predictions, it was shown that this can be compensated for by successively linearizing the pulse pattern around the modified switching instants (instead of always linearizing around the nominal switching instants).

Two controller variants were proposed: the standard and advanced controllers. The latter makes use of successive linearization to accurately predict the state vector, whereas this is absent from the former controller, resulting in inaccurate state predictions but with a decrease in the computational burden. The advanced controller displayed a short response time during the dynamic operation of a converter system. The standard controller, although not as fast as its advanced counterpart, exhibited a decent response.

Finally, an upper bound was established for the small-signal controller by comparing its response to a nonlinearized controller. It was seen that the small-signal controller performs in a similar manner to the nonlinearized controller which has a significantly higher computational burden.

Chapter 6

Constrained Small-Signal Controller

In this chapter, the formulation of the small-signal controller is extended to include constraints on the state vector. Specifically, constraints, representing the bounds that the state variables should stay within, are added to the optimization problem. First, the bound types are discussed. The bounds are then formulated as constraints and added to the QP underlying the small-signal controller. Finally, the performance of the constrained small-signal controller is evaluated during a start-up transient, and the results are discussed.

Chapter Contents

6.1 The State Constraints Problem

In Section 5.5.3, constraints are placed on the strength vector $\boldsymbol{\Lambda}$ (which is the decision variable of the optimization problem) in order to maintain a feasible pulse pattern. However, further constraints are required on the state vector $\boldsymbol{x}$ to ensure the converter states are kept within certain bounds; these bounds typically represent a safe operating region for the converter system. These additional constraints are added to the underlying QP [see (5.54)] of the control algorithm. Note that only the advanced controller is considered in this chapter, since the standard controller is simply a reduced version thereof.

As a case study, the (higher-order) grid-connected converter system from Section 3.3.2 is used. Recall that the state vector consists of the converter currents, grid currents, and capacitor voltages in the $\alpha\beta$ reference frame,

$$\boldsymbol{x}(t) = \begin{bmatrix} i_\alpha(t) & i_\beta(t) & i_{g,\alpha}(t) & i_{g,\beta}(t) & v_{c,\alpha}(t) & v_{c,\beta}(t) \end{bmatrix}^{\mathrm{T}}.$$

6.2 Constrained Small-Signal Controller

In this section, the formulation of the constrained small-signal controller is derived. In Section 6.2.1, the bounds are discussed, which are then formulated as constraints in Section 6.2.2. During the formulation of the constraints, additional optimization variables are introduced that can temporarily increase the limits of the bounds in order to ensure feasibility during optimization. Finally, in Section 6.2.3, the QP of (5.54) is augmented with the additional constraints.

6.2.1 Selecting the Bound

Although directly applying the bounds in the $\alpha\beta$ reference frame is convenient (and requires no additional transformations), the converter limits are usually defined in terms of the three-phase variables as

$$|\xi_a| \leq \sigma_\xi, \ |\xi_b| \leq \sigma_\xi, \text{ and } |\xi_c| \leq \sigma_\xi,$$

where σ_ξ is the limit on a specific converter quantity. Note that the bounds are assumed to be symmetrical and all components of a specific quantity have the same limit. By using the reduced inverse Clarke transformation of (3.5), the bounds that are defined in the three-phase plane can be mapped to the $\alpha\beta$ reference plane as

$$\left|\boldsymbol{K}^{-1}\boldsymbol{\xi}_{\alpha\beta}\right| \leq \mathbf{1}_3\sigma_\xi, \tag{6.1}$$

where the absolute value is componentwise, and $\mathbf{1}_n$ is an n-dimensional vector of ones. The resulting shape of the bounds is a hexagon, as shown in Figure 6.1. Recall that the reduced Clarke transformation assumes that the common-mode components of the transformed variables are zero, which is fortunately true in the case of the grid-connected converter of Section 3.3.2. In the case of nonzero common-mode components, the common-mode components have to be identified and the full inverse Clarke transformation has to be used. Alternatively, the control problem can be exclusively formulated in the three-phase reference frame.

For brevity, other bounds are not discussed since they either result in nonlinear constraints (which significantly increase the complexity of the optimization problem), or are suboptimally defined (for example, if defined directly in the $\alpha\beta$ reference frame). The interested reader is referred to [4, Section 11.1.3] for further reading on other bounds.

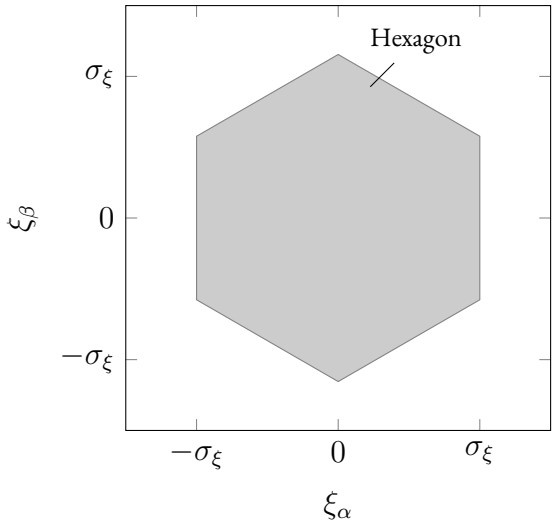


Figure 6.1: Bounds mapped from *abc*.

With the grid-connected converter as a case study, denote with σ_i, σ_{i_g}, and σ_{v_c} the limits on the converter currents, grid currents, and capacitor voltages, respectively. For now, it is assumed that all converter states have limits. By applying (6.1) to all of the converter states, the resulting bounds are

$$\left|\boldsymbol{K}_{\text{aug}}^{-1}\boldsymbol{x}(t)\right| \leq \boldsymbol{S}\boldsymbol{\sigma} \tag{6.2}$$

where $\boldsymbol{K}_{\text{aug}}^{-1} \in \mathbb{R}^{9\times 6}$ is introduced as the block diagonal matrix with $\boldsymbol{K}^{-1}$ repeated on its diagonal,

$$\boldsymbol{K}_{\text{aug}}^{-1} = \begin{bmatrix} \boldsymbol{K}^{-1} & & \\ & \boldsymbol{K}^{-1} & \\ & & \boldsymbol{K}^{-1} \end{bmatrix},$$

$\boldsymbol{\sigma} \in \mathbb{R}^3$ is introduced as the vector containing the limits (or bounds),

$$\boldsymbol{\sigma} = \begin{bmatrix} \sigma_i & \sigma_{i_g} & \sigma_{v_c} \end{bmatrix}^{\text{T}}, \tag{6.3}$$

and $\boldsymbol{S} \in \mathbb{R}^{9\times 3}$ is defined as the block diagonal matrix with $\mathbf{1}_3$ repeated on its diagonal,

$$\boldsymbol{S} = \begin{bmatrix} \mathbf{1}_3 & & \\ & \mathbf{1}_3 & \\ & & \mathbf{1}_3 \end{bmatrix}.$$

6.2.2 Formulating the Constraints

Ideally, the state vector should be evaluated, and ensured to be within the bounds, across the entire continuous-time prediction horizon,

$$\left|\boldsymbol{K}_{\text{aug}}^{-1}\boldsymbol{x}(t)\right| \leq \boldsymbol{S}\boldsymbol{\sigma} \quad \text{for all } t \in [0, T_p].$$

Unfortunately, this is difficult to realize efficiently. Even just calculating where the peak[1] of the state vector occurs is a demanding task due to the multiple stationary points in the state vector prediction

[1]Define the *peak* value of the state vector as the maximum value of the state vector over the prediction interval, $\max_{0\leq t\leq T_p}\left|\boldsymbol{K}_{\text{aug}}^{-1}\boldsymbol{x}(t)\right|$.

(see Figure 5.12), as there is no simple closed-form expression that returns these points. Even if the peak value has been calculated, it is demanding yet again to dynamically incorporate it within the optimization procedure; the peak value and the point in time at which it occurs will be a function of the decision variable. Instead, the state vector is predicted at discrete-time instants across the prediction horizon as

$$\left|\boldsymbol{K}_{\text{aug}}^{-1}\boldsymbol{x}(kT_c)\right| \leq \boldsymbol{S}\boldsymbol{\sigma} \quad \text{for all } k = 1, 2, \ldots, N_c, \tag{6.4}$$

where T_c is the *constraint prediction interval* (that is, the time interval between state-constraint prediction instants) and $N_c = \frac{T_p}{T_c}$ is the number of state-constraint predictions.

The direct implementation of the (hard) constraints of (6.4) is naive and optimistic, as infeasibility is likely to occur at instances where the state vector $\boldsymbol{x}$ cannot be contained within its bounds; the optimization algorithm will return an infeasible solution. This may occur due to inaccurate predictions, measurement errors, faults that cause the state vector to move outside the bounds, or if the limits of the bounds are too tight. One (naive) option, if infeasibility occurs, is to repeat the optimization problem with the bounds removed (or increased). This approach is computationally inefficient, and, if the bounds are completely removed, does not actively drive the state vector back towards the bounds. Fortunately, it is possible to introduce additional decision variables that can temporarily relax the (limits of the) bounds *during* optimization to ensure feasibility. This gives rise to so-called *soft* constraints. Introduce with

$$\Delta\boldsymbol{\sigma}_k = \begin{bmatrix}\Delta\sigma_{i,k} & \Delta\sigma_{i_g,k} & \Delta\sigma_{v_c,k}\end{bmatrix}^{\mathrm{T}} \tag{6.5}$$

the *relaxation vector* at the kth state-constraint prediction instant, where $\Delta\sigma_{i,k}$, $\Delta\sigma_{i_g,k}$, and $\Delta\sigma_{v_c,k}$ are the relaxation (or slack) variables on the converter currents, grid currents, and capacitor voltages, respectively. The relaxation vectors are applied to (6.4), resulting in the bounds

$$\left|\boldsymbol{K}_{\text{aug}}^{-1}\boldsymbol{x}(kT_c)\right| \leq \boldsymbol{S}(\boldsymbol{\sigma} + \Delta\boldsymbol{\sigma}_k) \quad \text{for all } k = 1, 2, \ldots, N_c \tag{6.6}$$

that can be relaxed.

Next, the constraints of (6.6) are written in a compact matrix form. Recall that the small-signal error (for the advanced controller) is described as

$$\tilde{\boldsymbol{x}}(t, \boldsymbol{\Lambda}) = \boldsymbol{e}^{\boldsymbol{F}t}\tilde{\boldsymbol{x}}_0 + \boldsymbol{\Gamma}(t) + \boldsymbol{\Phi}(t)\boldsymbol{\Lambda},$$

where (5.31) is repeated here for convenience. From the definition of (5.28), the predicted state vector follows as the sum of the small-signal error $\tilde{\boldsymbol{x}}$ and the steady-state trajectory $\boldsymbol{x}^*$,

$$\boldsymbol{x}(t, \boldsymbol{\Lambda}) = \boldsymbol{e}^{\boldsymbol{F}t}\tilde{\boldsymbol{x}}_0 + \boldsymbol{\Gamma}(t) + \boldsymbol{\Phi}(t)\boldsymbol{\Lambda} + \boldsymbol{x}^*(t). \tag{6.7}$$

By inserting (6.7) into (6.6), and using the fact that $|\xi| \leq \zeta \Leftrightarrow -\zeta \leq \xi \leq \zeta$, the kth state-constraint prediction instant results in two sets of inequality constraints,

$$\boldsymbol{K}_{\text{aug}}^{-1}\left(\boldsymbol{e}^{\boldsymbol{F}kT_c}\tilde{\boldsymbol{x}}_0 + \boldsymbol{\Gamma}(kT_c) + \boldsymbol{\Phi}(kT_c)\boldsymbol{\Lambda} + \boldsymbol{x}^*(kT_c)\right) \leq \boldsymbol{S}(\boldsymbol{\sigma} + \Delta\boldsymbol{\sigma}_k) \tag{6.8a}$$

$$\boldsymbol{K}_{\text{aug}}^{-1}\left(\boldsymbol{e}^{\boldsymbol{F}kT_c}\tilde{\boldsymbol{x}}_0 + \boldsymbol{\Gamma}(kT_c) + \boldsymbol{\Phi}(kT_c)\boldsymbol{\Lambda} + \boldsymbol{x}^*(kT_c)\right) \geq -\boldsymbol{S}(\boldsymbol{\sigma} + \Delta\boldsymbol{\sigma}_k). \tag{6.8b}$$

Following some basic algebraic manipulations, the kth state-constraint prediction instant of (6.8) in compact matrix form is

$$\begin{bmatrix}\boldsymbol{A}_{\sigma,k} & -\boldsymbol{S}_{\text{aug}}\end{bmatrix}\begin{bmatrix}\boldsymbol{\Lambda} \\ \Delta\boldsymbol{\sigma}_k\end{bmatrix} \leq \boldsymbol{b}_{\sigma,k},$$

where $\boldsymbol{A}_{\sigma,k} \in \mathbb{R}^{18\times n_{\text{sw}}}$, $\boldsymbol{b}_{\sigma,k} \in \mathbb{R}^{18}$, and $\boldsymbol{S}_{\text{aug}} \in \mathbb{R}^{18\times 3}$ are defined as

$$\boldsymbol{A}_{\sigma,k} = \begin{bmatrix} \boldsymbol{K}_{\text{aug}}^{-1}\boldsymbol{\Phi}(kT_c) \\ -\boldsymbol{K}_{\text{aug}}^{-1}\boldsymbol{\Phi}(kT_c) \end{bmatrix}, \quad \boldsymbol{b}_{\sigma,k} = \begin{bmatrix} \boldsymbol{S}\boldsymbol{\sigma} - \boldsymbol{\nu}(kT_c) \\ \boldsymbol{S}\boldsymbol{\sigma} + \boldsymbol{\nu}(kT_c) \end{bmatrix}, \text{ and } \boldsymbol{S}_{\text{aug}} = \begin{bmatrix} \boldsymbol{S} \\ \boldsymbol{S} \end{bmatrix}, \tag{6.9}$$

respectively, and $\boldsymbol{\nu} \in \mathbb{R}^9$ is introduced as

$$\boldsymbol{\nu}(kT_c) = \boldsymbol{K}_{\text{aug}}^{-1}\left(e^{\boldsymbol{F}kT_c}\tilde{\boldsymbol{x}}_0 + \boldsymbol{\Gamma}(kT_c) + \boldsymbol{x}^*(kT_c)\right).$$

Each state-constraint prediction instant results in 18 linear constraints and 3 additional decision variables that are added to the optimization problem: there are nine states (in abc), and each state requires two linear constraints to realize its bounds. Note that it is not required to constrain all of the converter states. The augmented strength vector $\boldsymbol{\Lambda}_{\text{aug}} \in \mathbb{R}^{n_{\text{sw}}+3N_c}$ is introduced as

$$\boldsymbol{\Lambda}_{\text{aug}} = \begin{bmatrix} \boldsymbol{\Lambda}^{\text{T}} & \Delta\boldsymbol{\sigma}_1^{\text{T}} & \Delta\boldsymbol{\sigma}_2^{\text{T}} & \cdots & \Delta\boldsymbol{\sigma}_{N_c}^{\text{T}} \end{bmatrix}^{\text{T}}, \tag{6.10}$$

where the original decision variables (the strength vector $\boldsymbol{\Lambda}$) have been augmented with the N_c relaxation vectors. By aggregating all the state-constraint prediction instants, for $k = 1, 2, \ldots, N_c$, the full constraints in matrix form are

$$\boldsymbol{A}_\sigma \boldsymbol{\Lambda}_{\text{aug}} \leq \boldsymbol{b}_\sigma, \tag{6.11}$$

where $\boldsymbol{A}_\sigma \in \mathbb{R}^{18N_c\times(n_{\text{sw}}+3N_c)}$ and $\boldsymbol{b}_\sigma \in \mathbb{R}^{18N_c}$ are defined as

$$\boldsymbol{A}_\sigma = \begin{bmatrix} \boldsymbol{A}_{\sigma,1} & \boldsymbol{S}_{\text{aug}} & \boldsymbol{0}_{18\times 3} & \cdots & \boldsymbol{0}_{18\times 3} \\ \boldsymbol{A}_{\sigma,2} & \boldsymbol{0}_{18\times 3} & \boldsymbol{S}_{\text{aug}} & \cdots & \boldsymbol{0}_{18\times 3} \\ \vdots & & & \ddots & \\ \boldsymbol{A}_{\sigma,N_c} & \boldsymbol{0}_{18\times 3} & \boldsymbol{0}_{18\times 3} & \cdots & \boldsymbol{S}_{\text{aug}} \end{bmatrix} \quad \text{and} \quad \boldsymbol{b}_\sigma = \begin{bmatrix} \boldsymbol{b}_{\sigma,1} \\ \boldsymbol{b}_{\sigma,2} \\ \vdots \\ \boldsymbol{b}_{\sigma,N_c} \end{bmatrix},$$

respectively.

6.2.3 Augmented Optimization Problem

The QP of (5.54) is augmented with the (state vector) constraints of (6.11) as

$$\boldsymbol{\Lambda}_{\text{aug,opt}} = \arg\min_{\boldsymbol{\Lambda}_{\text{aug}}} \quad \tfrac{1}{2}\boldsymbol{\Lambda}_{\text{aug}}^{\text{T}}\boldsymbol{H}_{\text{aug}}\boldsymbol{\Lambda}_{\text{aug}} + \boldsymbol{c}_{\text{aug}}^{\text{T}}\boldsymbol{\Lambda}_{\text{aug}} \tag{6.12a}$$

$$\text{subject to} \quad \begin{bmatrix} \boldsymbol{A} & \boldsymbol{0}_{(n_{\text{sw}}+3)\times 3N_c} \\ & \boldsymbol{A}_\sigma \end{bmatrix} \boldsymbol{\Lambda}_{\text{aug}} \leq \begin{bmatrix} \boldsymbol{b} \\ \boldsymbol{b}_\sigma \end{bmatrix}, \tag{6.12b}$$

where the augmented Hessian $\boldsymbol{H}_{\text{aug}} \in \mathbb{R}^{(n_{\text{sw}}+3N_c)\times(n_{\text{sw}}+3N_c)}$ and vector $\boldsymbol{c} \in \mathbb{R}^{(n_{\text{sw}}+3N_c)}$ are

$$\boldsymbol{H}_{\text{aug}} = \begin{bmatrix} \boldsymbol{H} & \\ & \gamma\mathbf{I}_{3N_c} \end{bmatrix} \quad \text{and} \quad \boldsymbol{c}_{\text{aug}} = \begin{bmatrix} \boldsymbol{c} \\ \boldsymbol{0}_{3N_c} \end{bmatrix},$$

respectively, where $\gamma \in \mathbb{R}$ is the nonnegative *relaxation weight* that penalizes relaxation (of the bounds). Note that the original constraint matrix $\boldsymbol{A}$, defined in Section 5.5.3, is augmented with $\boldsymbol{0}_{(n_{\text{sw}}+3)\times 3N_c}$ since it is not a function of the relaxation vectors $\Delta\boldsymbol{\sigma}_k$. It can be deduced that all elements of the relaxation vector will only be positive (that is, the bounds are never decreased) and will be zero if the bounds do not have to be increased.

Relaxation Weight Selection

A large relaxation weight γ implies that the controller relaxes the bounds as little as possible since $\Delta\boldsymbol{\sigma}_k$ is penalized significantly, whereas a small relaxation weight γ results in the controller exerting little effort to keep the state vector within the bounds since $\Delta\boldsymbol{\sigma}_k$ can be increased freely. At first, it seems sensible to choose a very large relaxation weight γ so that the bounds increase *just enough* to maintain feasibility. However, this is known to lead to ill-conditioning. To demonstrate this, recall that the conditioning number of the original Hessian $\boldsymbol{H}$ is

$$\kappa = \frac{L_c}{\mu} \geq 1,$$

where L_c (the tight Lipschitz constant) and μ (the tight convexity parameter) are the largest and smallest eigenvalues of the Hessian $\boldsymbol{H}$, respectively. Furthermore, note that the eigenvalues of the augmented Hessian $\boldsymbol{H}_{\text{aug}}$ are the relaxation weight γ and those of the original Hessian $\boldsymbol{H}$. Thus, if addition of state constraints should not influence the conditioning of the augmented Hessian $\boldsymbol{H}_{\text{aug}}$, and therefore the convergence of the gradient method, the relaxation weight γ should be in the interval $[\mu, L_c]$. If the relaxation weight γ is not in this interval, it increases the largest, or decreases the smallest, eigenvalue of the augmented Hessian $\boldsymbol{H}_{\text{aug}}$, which increases the conditioning number. Since it is beneficial to have the weight as large as possible (to relax the bounds as little as possible), the relaxation weight γ should be equal to the largest eigenvalue of the original Hessian $\boldsymbol{H}$, $\gamma = L_c$. Note that the relaxation weight γ can never improve conditioning.

6.3 Performance Evaluation

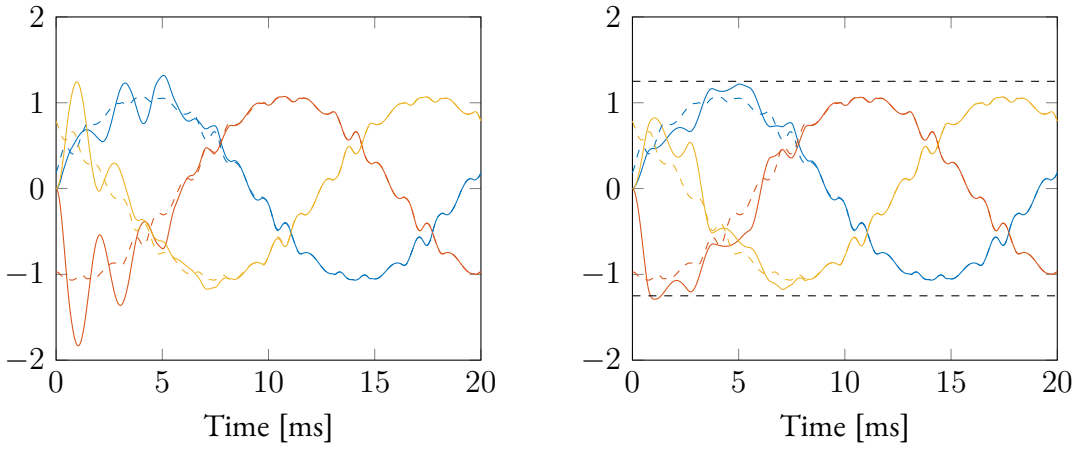
To evaluate the performance of the constrained small-signal controller, the converter system is considered during a start-up transient (when all the converter states are initially zero).[2] The parameters of the converter system can be found in Section 3.3.2. The prediction horizon and sampling interval of the controller are set to $T_p = 2\,\text{ms}$ and $T_s = 25\,\mu\text{s}$, respectively. A pulse number of $d = 5$ is selected, resulting in a switching frequency of $f_{\text{sw}} = 250\,\text{Hz}$. All state variables are penalized equally with $\boldsymbol{Q} = \mathbf{I}_6$ and all switching transition modifications are also penalized equally with $\boldsymbol{R} = \mathbf{I}_{n_{\text{sw}}}$. Unless mentioned otherwise, the constraint prediction interval is set to $T_c = 200\,\mu\text{s}$, resulting in $N_c = 10$ state-constraint prediction instants across the horizon of $T_p = 2\,\text{ms}$. At all instants, the relaxation weight is set equal to the largest eigenvalue of the Hessian $\boldsymbol{H}$, $\gamma = L_c$. No limits are imposed on the converter and grid currents, as they do not exhibit any significant overshoot during start-up. This implies that the vector containing the bounds and the relaxation vector reduce to (scalar) variables, that is, $\boldsymbol{\sigma} = \sigma_{v_c}$ and $\Delta\boldsymbol{\sigma}_k = \Delta\sigma_{v_c,k}$. This reduces the size of the optimization problem, since the dimension of the additional decision variables is reduced by a factor of three.

6.3.1 Multiple Relaxation Variables

The capacitor voltage during a start-up transient is shown in Figure 6.2. In the case that no bounds are imposed, it can be observed from Figure 6.2a that the peak[3] capacitor voltage is 1.8 pu. Once a tight bound of $\sigma_{v_c} = 1.25\,\text{pu}$ is placed on the capacitor voltage, it is seen in Figure 6.2b that the peak capacitor voltage reduces to 1.26 pu. It can be observed that bounds are slightly violated, implying that some relaxation is present.

[2]Note that a start-up transient is not a typical occurrence in practice, since (practical) converters are not started in this manner. However, for the purpose of benchmarking the controller, this case does present a useful case study, since it results in a significant overshoot in the capacitor voltage.

[3]Recall that the peak refers to the maximum absolute value.



(a) No capacitor voltage bounds.

(b) Capacitor voltage bounds of $\sigma_{v_c} = 1.25$ pu.

Figure 6.2: The capacitor voltage (in pu) during a start-up transient when using multiple relaxation variables. The bounds are indicated with black dashed lines.

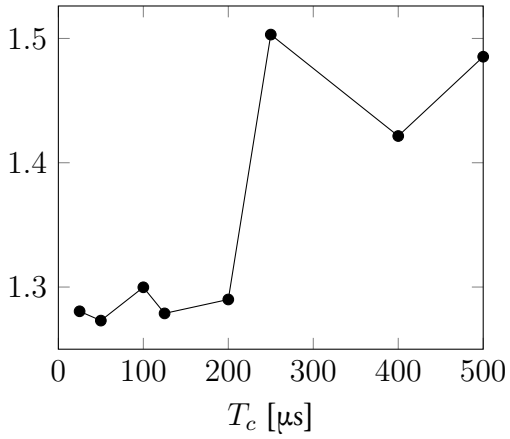


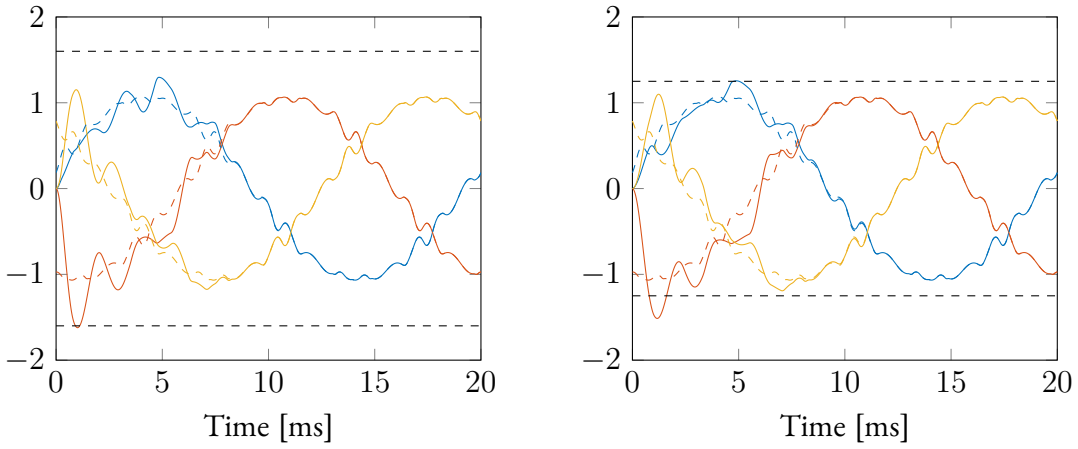
Figure 6.3: The peak capacitor voltage (in pu) as a function of the constraint interval T_c when using multiple relaxation variables for capacitor voltage bounds of $\sigma_{v_c} = 1.25$ pu.

In Figure 6.3, the peak capacitor voltage as a function of the constraint interval T_c is shown. As seen, in general, shorter constraint intervals T_c allow the controller to more accurately impose the bounds. However, it is noted that at some instances a longer constraint interval T_c performed better than a shorter interval. The cause of this is not yet known, but it may be attributed to inaccurate predictions. Recall that the predictions of the advanced controller only become accurate after the successive re-linearization around the modified switching instants; initially, the advanced controller tends to predict rather optimistic behaviour (see Figure 5.12).

6.3.2 Single Relaxation Variable

Now, the case of a single relaxation variable is considered. Specifically, only a single relaxation variable is used across the prediction horizon T_p. This significantly reduces the size of the optimization problem, as the dimension of the additional decision variables is reduced by a factor of N_c. However, having a single relaxation variable results in when the bounds are relaxed, they are relaxed by the same amount across the entire prediction horizon T_p (which most likely reduces the performance of the controller).

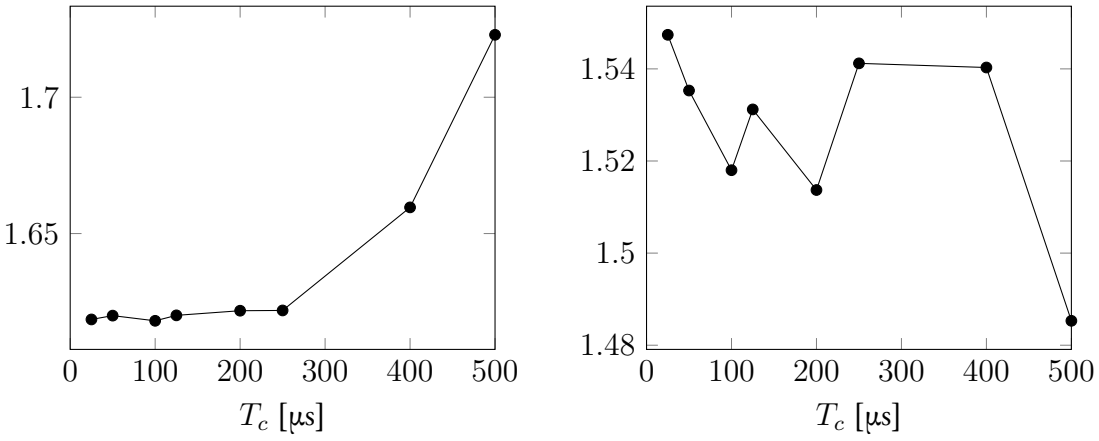
First, a (relatively) loose bound of $\sigma_{v_c} = 1.6$ pu is placed on the capacitor voltage. As seen in



(a) Capacitor voltage bounds of $\sigma_{v_c} = 1.6$ pu.

(b) Capacitor voltage bounds of $\sigma_{v_c} = 1.25$ pu.

Figure 6.4: The capacitor voltage (in pu) during a start-up transient when using a single relaxation variable. The bounds are indicated with the black dashed lines.



(a) Capacitor voltage bounds of $\sigma_{v_c} = 1.6$ pu.

(b) Capacitor voltage bounds of $\sigma_{v_c} = 1.25$ pu.

Figure 6.5: The peak capacitor voltage (in pu) as a function of the constraint interval T_c when using a single relaxation variable.

Figure 6.4a, the capacitor voltage reached a peak value of 1.61 pu. The bound is then significantly tightened to $\sigma_{v_c} = 1.25$ pu. As seen in Figure 6.4b, the bounds are poorly met, with the peak capacitor voltage only reducing to 1.55 pu. Furthermore, increasing the relaxation weight γ does not remedy the poor adherence to the bounds. Note the decrease in performance when compared to using multiple relaxation variables (see Figure 6.2b).

Figure 6.5 illustrates the effect that the constraint interval T_c has on the peak capacitor voltage. In Figure 6.5a, it can be observed that for a (loose) bound of $\sigma_{v_c} = 1.6$ pu, the length of the constraint intervals T_c resulted in expected behaviour: a finer constraint interval results in better adherence to the bounds. However, when a very tight bound of $\sigma_{v_c} = 1.25$ pu is selected, the performance becomes unpredictable as shown in Figure 6.5b.

6.4 Summary

A method to impose bounds on the state vector was proposed in this chapter. Owing to the shape of the bounds, the constraints are linear and can easily be added to the QP underlying the small-

signal controller. Relaxation variables were introduced that gave rise to so-called soft constraints, which allowed the optimization problem to always maintain feasibility.

The performance evaluation of the constrained small-signal controller was analyzed when using a single relaxation variable and multiple relaxation variables. When using a single relaxation variable, the method works relatively well given that the bounds were not too tight. In the case of using multiple relaxation variables, the method worked well, even if tight bounds were selected.

Chapter 7

Control of Neutral-Point Potential

In this chapter, balancing of the neutral-point potential is integrated in the standard small-signal controller. First, a review of the neutral-point potential control problem is given, including an overview of the traditional methods that are used. Then, a method to determine the steady-state trajectory, which includes the effect of the neutral-point potential, of a converter system that is modulated by OPPs is presented. Thereafter, it is shown how modifications to the absolute value of a pulse pattern can be efficiently modelled. Finally, the small-signal controller with integrated balancing of the neutral-point potential is derived. The performance of the controller is evaluated, by means of simulation, during transients as well as during conditions where traditional methods fail. The results are discussed.

Chapter Contents

7.1 The Neutral-Point Potential Control Problem

Recall from Section 3.2.2 that the evolution of the neutral-point potential of the NPC converter is described by

$$\frac{\mathrm{d}v_n(t)}{\mathrm{d}t} = \frac{1}{2C_d}\left(|u_a(t)|\, i_a(t) + |u_b(t)|\, i_b(t) + |u_c(t)|\, i_c(t)\right), \tag{7.1}$$

where (3.14) is repeated here for convenience. As evident from (7.1), the differential equation that describes the neutral-point potential v_n is nonlinear, as it involves the products of state variables (specifically, the three-phase converter current $\boldsymbol{i}_{abc}$) and the inputs (in this case, the three-phase pulse pattern $\boldsymbol{u}_{abc}$). The NPC converter exhibits so-called *natural balancing properties* [60]; under certain circumstances, the neutral-point potential v_n will balance automatically and its average (dc) value will be zero in steady-state conditions. It is shown in [60] that the load impedance and the spectra characteristics of the switching signal $\boldsymbol{u}_{abc}$ (that is, the switch positions over the fundamental period) determine the effectiveness of the natural balancing mechanisms. Specifically, if the sets of spectra of (the switching signals) u_a and $|u_a|$, u_b and $|u_b|$, and u_c and $|u_c|$ are all orthogonal (meaning, they do not overlap), then the neutral-point potential v_n is balanced in the steady state. It is relatively easy to show that for a (nominal) quarter-wave symmetrical pulse pattern that these spectra are orthogonal. However, the natural balancing properties that certain pulse patterns possess can be weak and easily disrupted when a controller modifies the pulse pattern to achieve closed-loop control. This may inadvertently lead to a loss of natural balancing and, consequently, the drifting away of the neutral-point potential v_n; thus, it may be required to actively balance the neutral-point potential via a control method.

Traditionally, the control methods of the neutral-point potential v_n can be categorized into two groups. The first group of methods, which has only been proposed for CB-PWM, uses the principle of common-mode injection by manipulating the common-mode component u_0 of the switching signal $\boldsymbol{u}_{abc}$. This method was first introduced in [61], and then later extended in [62] and [63]. The common-mode component u_0 induces a current in neutral point, which in turn modifies the neutral-point potential v_n. Common-mode injection is typically achieved by having an outer control loop inject the appropriate common-mode term u_0 into the modulating signal that is fed to the modulator. However, the effectiveness of common-mode injection diminishes as the phase between the converter voltage and current approaches 90 degrees (see [63, (9)]).

The second group of methods exploits the redundant vectors of the NPC converter, which was first proposed in [64]. Recall from Section 3.2.1 that there are six pairs of three-phase switch positions $\boldsymbol{u}_{abc}$ that result in the same differential-mode voltages $\boldsymbol{v}_{\alpha\beta}$ but with opposite common-mode voltages v_0; the common-mode voltages have an opposite effect on the neutral-point potential v_n. Thus, one of the vectors of the redundant pair will increase the neutral-point potential whereas the other one will decrease it. Note that, similar to the principle of common-mode injection, the method of redundant vectors is also a form of common-mode voltage v_0 manipulation and is not effective when the phase between the converter voltage and current is 90 degrees (that is, when the power factor at the converter terminals is zero).

Literature of neutral-point potential balancing methods for a converter that is modulated by OPPs is very limited. The first neutral-point potential control method for pulse patterns was presented in [65], where the method of redundant vectors was extended to pulse patterns. In [66], a neutral-point potential control method was integrated in the MP^{3}C algorithm. An additional term was added to the objective function that penalizes the dc-component of the neutral-point potential v_n. In order to target only the dc-component, a low-pass filter is required. A number of assumptions were made during the derivation of the control method that proved to be detrimental to the performance. First, the converter current is assumed to be purely sinusoidal. Second, modi-

fications to the converter current are not taken into consideration. Finally, the converter current is assumed to be constant between the switching transitions of the pulse pattern. These assumptions effectively reduced the method to a form of common-mode injection, and thus the method is not effective under zero power factor at the converter terminals. In order to achieve balancing of the neutral-point potential v_n at all operating conditions, a control method should consider the ripple of the converter current; a common-mode injection-only approach is insufficient.

In this chapter, balancing of the neutral-point potential is integrated in the standard small-signal controller (from Section 5.5.7). Unlike the aforementioned balancing methods, the proposed method does not rely on manipulating the common-mode voltage v_0 of the converter, and is thus effective across all operating conditions (including zero power factor at the converter terminals). In accordance with the control principle of the small-signal controller, the neutral-point potential balancing method regulates the (instantaneous) neutral-point potential v_n along its steady-state trajectory v_n^*; no additional filters are required to extract the dc-component. Note that this method does not reduce the ripple of the neutral-point potential v_n in the steady state, since it is a natural characteristic during steady-state operation.

As a case study to demonstrate the balancing method, a first-order converter system with parameters of Table 3.5 and a half dc-link capacitance of $C_d = 3.43\,\text{pu}$ is considered. The derivation of the balancing method is general enough so that any load can be considered. When the effect of the neutral-point potential v_n is taken into account, the three-phase output voltage of an NPC converter is

$$\boldsymbol{v}_{abc}(t) = \frac{V_d}{2}\boldsymbol{u}_{abc}(t) - v_n(t)\,|\boldsymbol{u}_{abc}(t)|\,,$$

where the absolute value is componentwise, $|\boldsymbol{u}_{abc}| = [|u_a| \;\; |u_b| \;\; |u_c|]^{\mathrm{T}}$. Note that the neutral-point potential v_n is also a state variable of the system. However, for convenience in subsequent sections, the state vector $\boldsymbol{x} \in \mathbb{R}^{n_x}$ is used for all converter states other than the neutral-point potential v_n, whereas the so-called *complete* state vector $\boldsymbol{x}_{\text{np}} \in \mathbb{R}^{n_x+1}$ includes the neutral-point potential as

$$\boldsymbol{x}_{\text{np}}(t) = \begin{bmatrix}\boldsymbol{x}^{\mathrm{T}}(t) & v_n(t)\end{bmatrix}^{\mathrm{T}}. \tag{7.2}$$

In the case of a first-order converter system, the state vector is the converter currents in the $\alpha\beta$ reference frame,

$$\boldsymbol{x}(t) = \begin{bmatrix}i_\alpha(t) & i_\beta(t)\end{bmatrix}^{\mathrm{T}}.$$

Furthermore, the evolution of the neutral-point potential in (7.1) can be expressed as

$$\frac{\mathrm{d}v_n(t)}{\mathrm{d}t} = \frac{1}{2C_d}\left(|\boldsymbol{u}_{abc}(t)|^{\mathrm{T}}\boldsymbol{T}\boldsymbol{x}(t)\right), \tag{7.3}$$

where the transformation matrix $\boldsymbol{T} \in \mathbb{R}^{n_u \times n_x}$, introduced for generality, calculates the three-phase converter current $\boldsymbol{i}_{abc}$ from the state vector $\boldsymbol{x}$. In the case of the first-order converter system, the transformation matrix $\boldsymbol{T}$ is simply the inverse Clarke transformation matrix $\boldsymbol{K}^{-1}$. The (complete) converter system is described using the continuous-time state-space representation

$$\frac{\mathrm{d}\boldsymbol{x}_{\text{np}}(t)}{\mathrm{d}t} = \boldsymbol{F}_{\text{np}}(|\boldsymbol{u}_{abc}(t)|)\boldsymbol{x}_{\text{np}}(t) + \boldsymbol{G}_{\text{np}}\boldsymbol{u}_{abc}(t) + \boldsymbol{P}_{\text{np}}\boldsymbol{v}_g(t), \tag{7.4}$$

where $\boldsymbol{u}_{abc}$ is the three-phase pulse pattern, and $\boldsymbol{v}_g$ is the grid voltage in the $\alpha\beta$ reference frame. The (complete) state $\boldsymbol{F}_{\text{np}}(|\boldsymbol{u}_{abc}|) \in \mathbb{R}^{(n_x+1)\times(n_x+1)}$, input $\boldsymbol{G}_{\text{np}} \in \mathbb{R}^{(n_x+1)\times n_u}$, and disturbance

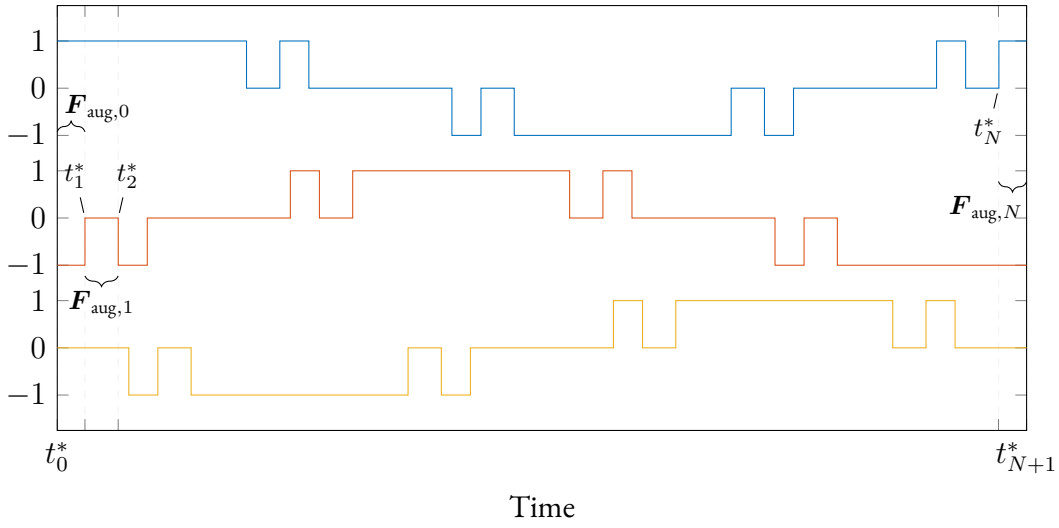


Figure 7.1: The different states that $\boldsymbol{F}_{\text{np}}(|\boldsymbol{u}_{abc}^*|)$ assumes over a fundamental period T_1.

$\boldsymbol{P}_{\text{np}} \in \mathbb{R}^{(n_x+1)\times n_v}$ matrices follow as

$$\begin{aligned} \boldsymbol{F}_{\text{np}}(|\boldsymbol{u}_{abc}(t)|) &= \begin{bmatrix} \boldsymbol{F} & \frac{1}{L}\boldsymbol{K}\,|\boldsymbol{u}_{abc}(t)| \\ \frac{1}{2C_d}\left(|\boldsymbol{u}_{abc}(t)|^{\mathrm{T}}\boldsymbol{T}\right) & 0 \end{bmatrix}, \\ \boldsymbol{G}_{\text{np}} &= \begin{bmatrix} \boldsymbol{G} \\ \boldsymbol{0}_{n_u}^{\mathrm{T}} \end{bmatrix}, \text{ and } \boldsymbol{P}_{\text{np}} = \begin{bmatrix} \boldsymbol{P} \\ \boldsymbol{0}_{n_v}^{\mathrm{T}} \end{bmatrix}, \end{aligned} \tag{7.5}$$

respectively, where the state-space matrices characterizing the load are

$$\boldsymbol{F} = -\frac{R}{L}\mathbf{I}_2,\ \boldsymbol{G} = \frac{V_d}{2L}\boldsymbol{K},\ \text{and}\ \boldsymbol{P} = -\frac{1}{L}\mathbf{I}_2.$$

7.2 Steady-State Trajectory Including the Neutral-Point Potential of a Converter System

The method from Section 5.3 used to determine the steady-state trajectory of a converter system that is modulated by the nominal pulse pattern $\boldsymbol{u}_{abc}^*$ can be extended to include the neutral-point potential v_n. Note that the state matrix $\boldsymbol{F}_{\text{np}}(|\boldsymbol{u}_{abc}^*|)$ is now time-varying and a function of the pulse pattern $\boldsymbol{u}_{abc}^*$. Specifically, $\boldsymbol{F}_{\text{np}}(|\boldsymbol{u}_{abc}^*|)$ is piecewise constant and assumes $N+1$, where $N = 12d$, different states during a fundamental period T_1. Denote with t_i^* the (three-phase) nominal switching times, where $i = 0, 1, \ldots, N+1$. The times $t_0^* = 0$ and $t_{N+1}^* = T_1$ are introduced to consider the boundaries. Define the state that $\boldsymbol{F}_{\text{np}}(|\boldsymbol{u}_{abc}^*|)$ assumes over a fundamental period T_1 with

$$\boldsymbol{F}_{\text{np},i} = \boldsymbol{F}_{\text{np}}(|\boldsymbol{u}_{abc}^*(t_i^*)|).$$

Figure 7.1 illustrates the (three-phase) nominal switching times t_i^* and the different states that $\boldsymbol{F}_{\text{np}}(|\boldsymbol{u}_{abc}^*|)$ assumes.

Unlike the steady-state trajectory that does not consider the effect of neutral-point potential, as described in (5.4), the steady-state trajectory $\boldsymbol{x}_{\text{np}}^*$ cannot be described by a single set of equations since the state matrix $\boldsymbol{F}_{\text{np},i}$ is piecewise constant. For example, the steady-state trajectory is described as

$$\boldsymbol{x}_{\text{np}}^*(t) = \boldsymbol{e}^{\boldsymbol{F}_{\text{np},0}t}\boldsymbol{x}_{\text{np},0}^* + \int_0^t \boldsymbol{e}^{\boldsymbol{F}_{\text{np},0}(t-\tau)}\boldsymbol{G}_{\text{np}}\boldsymbol{u}_{abc}^*(0)\,\mathrm{d}\tau + \int_0^t \boldsymbol{e}^{\boldsymbol{F}_{\text{np},0}(t-\tau)}\boldsymbol{P}_{\text{np}}\boldsymbol{v}_g(\tau)\,\mathrm{d}\tau$$

for time $t \in [0, t_1^*]$, and with

$$\boldsymbol{x}_{\text{np}}^*(t) = \boldsymbol{e}^{\boldsymbol{F}_{\text{np},1}(t-t_1^*)}\boldsymbol{x}_{\text{np}}^*(t_1^*) + \int_{t_1^*}^{t} \boldsymbol{e}^{\boldsymbol{F}_{\text{np},1}(t-\tau)}\boldsymbol{G}_{\text{np}}\boldsymbol{u}_{abc}^*(t_1^*)\,\mathrm{d}\tau + \int_{t_1^*}^{t} \boldsymbol{e}^{\boldsymbol{F}_{\text{np},1}(t-\tau)}\boldsymbol{P}_{\text{np}}\boldsymbol{v}_g(\tau)\,\mathrm{d}\tau$$

for time $t \in (t_1^*, t_2^*]$. Furthermore, unlike in Section 5.3, the effect of the grid voltage $\boldsymbol{v}_g$ cannot be calculated separately using simple phasor analysis due to the state matrix $\boldsymbol{F}_{\text{np},i}$ being piecewise constant. Recall that the grid voltage (vector) is defined as

$$\boldsymbol{v}_g(t) = \begin{bmatrix} v_{g,\alpha}(t) \\ v_{g,\beta}(t) \end{bmatrix} = \sqrt{\frac{2}{3}} V_{g,\text{LL}} \begin{bmatrix} \sin(\omega_1 t) \\ -\cos(\omega_1 t) \end{bmatrix}.$$

Although the integral involving the grid voltage $\boldsymbol{v}_g$, which contains products between the matrix exponential and trigonometric terms, is not particularly difficult to solve by using integration by parts, the solution is nonetheless tedious. Instead, by representing the grid voltage as a harmonic resonator,

$$\frac{\mathrm{d}\boldsymbol{v}_g(t)}{\mathrm{d}t} = \boldsymbol{F}_{\text{res}}\boldsymbol{v}_g(t)$$

where

$$\boldsymbol{F}_{\text{res}} = \omega_1 \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix},$$

the effect of the grid voltage $\boldsymbol{v}_g$ can be taken into account in a concise manner (without having to use integration by parts). The steady-state trajectory $\boldsymbol{x}_{\text{np}}^*$ is augmented with the grid voltage $\boldsymbol{v}_g$ (which is now an uncontrollable state variable) as

$$\boldsymbol{x}_{\text{aug}}^* = \begin{bmatrix} \boldsymbol{x}_{\text{np}}^{*\text{T}}(t) & \boldsymbol{v}_g^T(t) \end{bmatrix}^{\text{T}} \tag{7.6}$$

where the augmented state $\boldsymbol{F}_{\text{aug},i} \in \mathbb{R}^{(n_x+n_v+1)\times(n_x+n_v+1)}$ and input $\boldsymbol{G}_{\text{aug}} \in \mathbb{R}^{(n_x+n_v+1)\times n_u}$ matrices are

$$\boldsymbol{F}_{\text{aug},i} = \begin{bmatrix} \boldsymbol{F}_{\text{np},i} & \boldsymbol{P}_{\text{np}} \\ \boldsymbol{0}_{n_v\times(n_x+1)} & \boldsymbol{F}_{\text{res}} \end{bmatrix} \quad \text{and} \quad \boldsymbol{G}_{\text{aug}} = \begin{bmatrix} \boldsymbol{G}_{\text{np}} \\ \boldsymbol{0}_{n_v\times n_u} \end{bmatrix}, \tag{7.7}$$

respectively. The steady-state trajectory can be described by the following system of equations,

$$\boldsymbol{x}_{\text{aug}}^*(t) = \begin{cases} \boldsymbol{e}^{\boldsymbol{F}_{\text{aug},0}t}\boldsymbol{x}_{\text{aug},0}^* + \displaystyle\int_0^t \boldsymbol{e}^{\boldsymbol{F}_{\text{aug},0}(t-\tau)}\boldsymbol{G}_{\text{aug}}\boldsymbol{u}_{abc}^*(0)\,\mathrm{d}\tau & \text{for } t \in [0, t_1^*] \quad (7.8a) \\ \boldsymbol{e}^{\boldsymbol{F}_{\text{aug},1}(t-t_1^*)}\boldsymbol{x}_{\text{aug}}^*(t_1^*) + \displaystyle\int_{t_1^*}^t \boldsymbol{e}^{\boldsymbol{F}_{\text{aug},1}(t-\tau)}\boldsymbol{G}_{\text{aug}}\boldsymbol{u}_{abc}^*(t_1^*)\,\mathrm{d}\tau & \text{for } t \in (t_1^*, t_2^*] \quad (7.8b) \\ \vdots & \\ \boldsymbol{e}^{\boldsymbol{F}_{\text{aug},N}(t-t_N^*)}\boldsymbol{x}_{\text{aug}}^*(t_N^*) + \displaystyle\int_{t_N^*}^t \boldsymbol{e}^{\boldsymbol{F}_{\text{aug},N}(t-\tau)}\boldsymbol{G}_{\text{aug}}\boldsymbol{u}_{abc}^*(t_N^*)\,\mathrm{d}\tau & \text{for } t \in (t_N^*, T_1]. \quad (7.8c) \end{cases}$$

The initial steady-state value $\boldsymbol{x}_{\text{aug},0}^*$ can be calculated by exploiting the fact that

$$\boldsymbol{x}_{\text{aug}}^*(0) = \boldsymbol{x}_{\text{aug}}^*(T_1) \tag{7.9}$$

holds in steady-state conditions (due to periodicity). From (7.8a), it is easy to see that

$$\boldsymbol{x}_{\text{aug}}^*(0) = \mathbf{I}_{n_x}\boldsymbol{x}_{\text{aug},0}^*. \tag{7.10}$$

Note that in order to calculate

$$\boldsymbol{x}^*_{\text{aug}}(T_1) = \boldsymbol{e}^{\boldsymbol{F}_{\text{aug},N}(T_1-t^*_N)}\boldsymbol{x}^*_{\text{aug}}(t^*_N) + \int_{t^*_N}^{T_1} \boldsymbol{e}^{\boldsymbol{F}_{\text{aug},N}(T_1-\tau)}\boldsymbol{G}_{\text{aug}}\boldsymbol{u}^*_{abc}(t^*_N)\,\mathrm{d}\tau, \tag{7.11}$$

which follows from setting $t = T_1$ in (7.8c), all the preceding sets of equations must be calculated in ascending order; that is, (7.8a) is first calculated for $t = t^*_1$, then (7.8b) for $t = t^*_2$, and so forth until $\boldsymbol{x}^*_{\text{aug}}(t^*_N)$ is calculated. Successively inserting the preceding sets of equations into (7.11) results in

$$\boldsymbol{x}^*_{\text{aug}}(T_1) = \boldsymbol{B}_{\text{aug}}\boldsymbol{x}^*_{\text{aug},0} + \boldsymbol{f}_{\text{aug}}, \tag{7.12}$$

where

$$\boldsymbol{B}_{\text{aug}} = \prod_{i=0}^{N} \boldsymbol{e}^{\boldsymbol{F}_{\text{aug},i}(t^*_{i+1}-t^*_i)} \tag{7.13}$$

$$\begin{aligned}\boldsymbol{f}_{\text{aug}} = &\sum_{i=0}^{N-1}\left(\prod_{l=i+1}^{N} \boldsymbol{e}^{\boldsymbol{F}_{\text{aug},l}(t^*_{l+1}-t^*_l)} \int_{t^*_i}^{t^*_{i+1}} \boldsymbol{e}^{\boldsymbol{F}_{\text{aug},i}(t^*_{i+1}-\tau)}\boldsymbol{G}_{\text{aug}}\boldsymbol{u}^*_{abc}(t^*_i)\,\mathrm{d}\tau\right)\\ &+ \int_{t^*_N}^{T_1} \boldsymbol{e}^{\boldsymbol{F}_{\text{aug},N}(T_1-\tau)}\boldsymbol{G}_{\text{aug}}\boldsymbol{u}^*_{abc}(t^*_N)\,\mathrm{d}\tau.\end{aligned} \tag{7.14}$$

Note the order of the products of matrix exponentials are important, since they do not commute. Consider the integrals of (7.14), which have the form

$$\int_{t^*_i}^{t^*_{i+1}} \boldsymbol{e}^{\boldsymbol{F}_{\text{aug},i}(t^*_{i+1}-\tau)}\boldsymbol{G}_{\text{aug}}\boldsymbol{u}^*_{abc}(t^*_i)\,\mathrm{d}\tau.$$

These integrals are trivial to solve since $\boldsymbol{u}^*_{abc}$ is constant between the switching transitions t^*_i and t^*_{i+1}. Note that $\boldsymbol{F}_{\text{aug},i}$ is singular at instances where the neutral point does not conduct current. By using a theorem from [58] allows for the integral

$$\int_{t^*_i}^{t^*_{i+1}} \boldsymbol{e}^{\boldsymbol{F}_{\text{aug},i}(t^*_{i+1}-\tau)}\,\mathrm{d}\tau$$

to be calculated using the appropriate sub-matrix of

$$\boldsymbol{e}^{\begin{bmatrix}\boldsymbol{F}_{\text{aug},i} & \mathbf{I}_{(n_x+n_v+1)}\\ \mathbf{0}_{(n_x+n_v+1)\times 2(n_x+n_v+1)}\end{bmatrix}(t^*_{i+1}-t^*_i)} = \begin{bmatrix}\boldsymbol{e}^{\boldsymbol{F}_{\text{aug}}(t^*_{i+1}-t^*_i)} & \int_{t^*_i}^{t^*_{i+1}} \boldsymbol{e}^{\boldsymbol{F}_{\text{aug},i}(t^*_{i+1}-\tau)}\mathrm{d}\tau\\ \mathbf{0}_{(n_x+n_v+1)\times(n_x+n_v+1)} & \mathbf{I}_{(n_x+n_v+1)}\end{bmatrix},$$

where no assumption is made regarding the invertibility of $\boldsymbol{F}_{\text{aug},i}$.

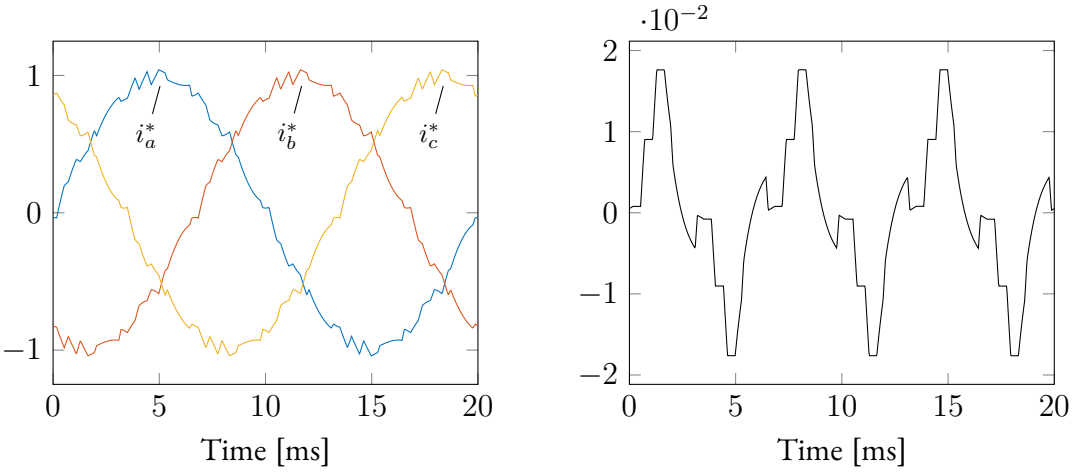
It follows from (7.9) [after inserting (7.10) and (7.12)] that

$$\boldsymbol{D}_{\text{aug}}\boldsymbol{x}^*_{\text{aug},0} = \boldsymbol{f}_{\text{aug}}$$

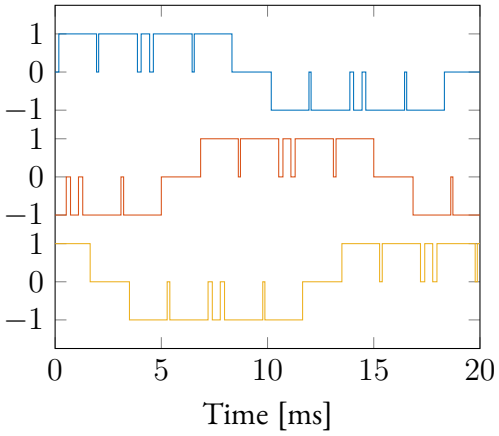
where

$$\boldsymbol{D}_{\text{aug}} = \left(\mathbf{I}_{(n_x+n_v+1)} - \boldsymbol{B}_{\text{aug}}\right).$$

Note that the initial augmented steady-state value $\boldsymbol{x}^*_{\text{aug},0}$ is not unique ($\boldsymbol{D}_{\text{aug}}$ is singular) since it depends on initial grid voltage $\boldsymbol{v}_{g,0}$; in a sense, the elements of $\boldsymbol{v}_{g,0}$ may be seen as free variables. However, the initial steady-state value $\boldsymbol{x}^*_{\text{np},0}$ can easily be determined given the initial grid voltage



(a) Converter currents (b) Neutral-point potential



(c) Switch positions

Figure 7.2: The steady-state trajectory $\boldsymbol{x}^*_{\text{np}}$ (in pu) of the converter system resulting from the nominal pulse pattern $\boldsymbol{u}^*_{abc}$.

$\boldsymbol{v}_{g,0}$. First, by knowing the grid voltage $\boldsymbol{v}_g$ is independent of the other state variables, and that for a harmonic resonator $e^{\boldsymbol{F}_{res}T_1} = \mathbf{I}_{n_v}$ holds true, it can be observed that $\boldsymbol{D}_{\text{aug}}$ and $\boldsymbol{f}_{\text{aug}}$ will have the forms

$$\boldsymbol{D}_{\text{aug}} = \begin{bmatrix} \boldsymbol{D}_{\text{np}} & \boldsymbol{D}_{\text{res}} \\ \mathbf{0}_{n_v \times (n_x+1)} & \mathbf{0}_{n_v \times n_v} \end{bmatrix} \quad \text{and} \quad \boldsymbol{f}_{\text{aug}} = \begin{bmatrix} \boldsymbol{f}_{\text{np}} \\ \mathbf{0}_{n_v} \end{bmatrix},$$

respectively. Then, by using basic manipulations, the initial steady-state value follows as

$$\boldsymbol{x}^*_{\text{np},0} = \boldsymbol{D}^{-1}_{\text{np}} \left(\boldsymbol{f}_{\text{np}} - \boldsymbol{D}_{\text{res}} \boldsymbol{v}_{g,0} \right), \tag{7.15}$$

where $\boldsymbol{v}_{g,0}$ is set to the appropriate initial grid voltage. With $\boldsymbol{x}^*_{\text{aug},0}$ calculated, the steady-state trajectory $\boldsymbol{x}^*_{\text{aug}}$ can be calculated over a fundamental period (at instants of T_s) using (7.8).

In Figure 7.2, an example of a steady-state trajectory $\boldsymbol{x}^*_{\text{np}}$ is shown. The pulse number is $d = 5$ and the system is operating at rated conditions with unity power factor.

7.3 Modelling Modification of the Absolute Value of the Pulse Pattern

Since the state matrix $\boldsymbol{F}_{\text{np}}(|\boldsymbol{u}_{abc}|)$ is a function of the absolute value of the pulse pattern, any modifications to the three-phase pulse pattern $\boldsymbol{u}_{abc}$ must also be reflected in its absolute value, $|\boldsymbol{u}_{abc}|$. To efficiently represent these modifications, the absolute value of the pulse pattern should be replaced with an expression that is more benign to use and manipulate. For the moment, only a single-phase pulse pattern is considered.

Recall from Section 5.4 that the incumbent pulse pattern is defined as

$$u(t) = u_0 + \sum_{i=1}^{n} \Delta u_i h(t - t_i'),$$

where $u_0 \in \{-1, 0, 1\}$, $\Delta u_i \in \{-1, 1\}$, and t_i' are the initial switch position, ith switching transition, and ith switching instant, respectively. Recall that only the standard controller from Section 5.5.7 is considered in this chapter, and thus the incumbent pulse pattern u is assumed to be equal to the nominal pulse pattern u^*. Furthermore, the modified pulse pattern is defined as

$$u_{\text{mod}}(t) = u_0 + \sum_{i=1}^{n} \Delta u_i h(t - (t_i' + \Delta t_i)),$$

where Δt_i is the time modification. The absolute value of the incumbent pulse pattern is represented by the so-called *absolute incumbent pulse pattern*,

$$u'(t) = |u(t)|\,, \tag{7.16}$$

which is described as

$$u'(t) = u_0' + \sum_{i=1}^{n} \Delta u_i' h(t - t_i') \tag{7.17}$$

where $u_0' \in \{0, 1\}$ and $\Delta u_i' \in \{-1, 1\}$ are the initial absolute switch position and ith absolute switching transition, respectively. Accordingly, the *absolute modified pulse pattern* is defined as

$$u_{\text{mod}}'(t) = |u_{\text{mod}}(t)|\,, \tag{7.18}$$

where

$$u_{\text{mod}}'(t) = u_0' + \sum_{i=1}^{n} \Delta u_i' h(t - (t_i' + \Delta t_i)). \tag{7.19}$$

The pulse patterns are shown in Figure 7.3.

7.3.1 Linear Approximation to Modifications of the Absolute Value of the Pulse Pattern

In Section 5.4.1, it is shown that the (linearized) modified pulse pattern is described as

$$u_{\text{mod}}(t) = u(t) + \tilde{u}(t),$$

where

$$\tilde{u}(t) = \sum_{i=1}^{n} \lambda_i \delta(t - t_i')$$

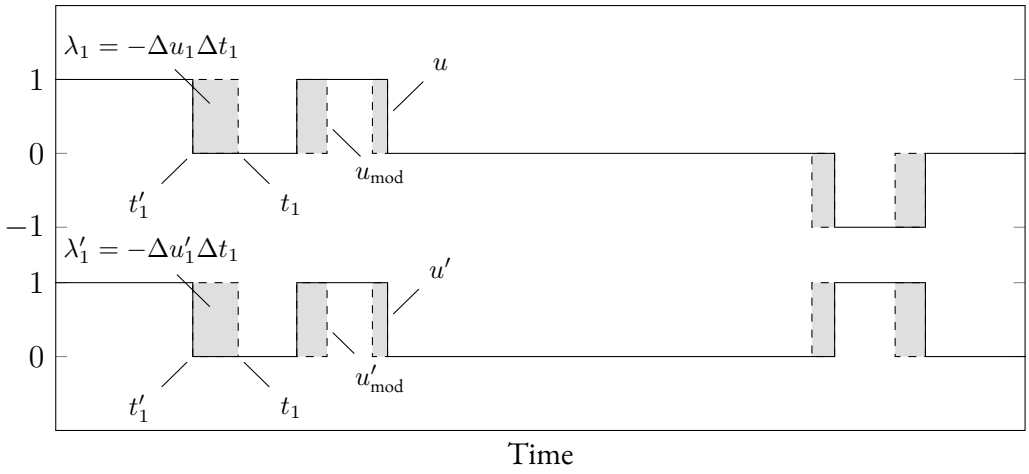


Figure 7.3: Representing the absolute value of the pulse pattern $|u|$ by the so-called absolute pulse pattern u'.

is the small-signal input consisting of impulses with strengths λ_i that occur at the incumbent switching instants t_i'. The ith impulse strength

$$\lambda_i = -\Delta u_i \Delta t_i$$

represents the ith rectangular area that is added or removed to the incumbent pulse pattern u (see Figure 7.3). By using a first-order Taylor series expansion of the step functions around the incumbent switching instants t_i', the linearization is now extended to the absolute modified pulse pattern as

$$u_{\text{mod}}'(t) = u'(t) + \tilde{u}'(t), \tag{7.20}$$

where

$$\tilde{u}'(t) = \sum_{i=1}^{n} \lambda_i' \delta(t - t_i') \tag{7.21}$$

is the *absolute small-signal input*. Similarly, the ith absolute impulse strength

$$\lambda_i' = -\Delta u_i' \Delta t_i \tag{7.22}$$

represents the ith rectangular area that is added or removed to the absolute incumbent pulse pattern u'. Since the time modifications Δt_i to the ith switching transition of the incumbent pulse pattern u and absolute incumbent pulse pattern u' are the same, it follows that the absolute impulse strength λ_i' can be written in terms of the impulse strength λ_i as

$$\lambda_i' = \frac{\Delta u_i'}{\Delta u_i} \lambda_i. \tag{7.23}$$

By inserting (7.23) into (7.21), the absolute small-signal input is expressed in terms of the impulse strengths λ_i as

$$\tilde{u}'(t) = \sum_{i=1}^{n} \frac{\Delta u_i'}{\Delta u_i} \lambda_i \delta(t - t_i'). \tag{7.24}$$

7.3.2 Three-Phase Case

This section generalizes the notation developed for the absolute pulse pattern in the previous section to the three-phase case. The modification of the ith switching instant of phase $p \in \{a, b, c\}$ is

$$\Delta t_{p,i} = t_{p,i} - t_{p,i}',$$

where $t_{p,i}$ and $t'_{p,i}$ are the modified and incumbent switching instants, respectively. The (direction of the) ith absolute switching transition in a particular phase is

$$\Delta u'_{p,i} = u'_{p,i} - u'_{p,i-1},$$

where $u'_{p,i}, u'_{p,i-1} \in \{0, 1\}$. By generalizing (7.22), the absolute impulse strength associated with the ith switching transition of phase p is

$$\lambda'_{p,i} = -\Delta u'_{p,i} \Delta t_{p,i},$$

which, according to (7.23), can also be expressed as

$$\lambda'_{p,i} = \frac{\Delta u'_{p,i}}{\Delta u_{p,i}} \lambda_{p,i}. \tag{7.25}$$

The (linearized) absolute modified pulse pattern of (7.20) is generalized to a three-phase absolute modified pulse pattern as

$$\boldsymbol{u}'_{abc,\text{mod}}(t, \lambda_{p,i}) = \boldsymbol{u}'_{abc}(t) + \tilde{\boldsymbol{u}}'_{abc}(t, \lambda_{p,i}) \tag{7.26}$$

where

$$\boldsymbol{u}'_{abc}(t) = \begin{bmatrix} u'_a(t) \\ u'_b(t) \\ u'_c(t) \end{bmatrix} = \begin{bmatrix} u'_{a,0} + \sum_{i=1}^{n_a} \Delta u'_{a,i} h(t - t'_{a,i}) \\ u'_{b,0} + \sum_{i=1}^{n_b} \Delta u'_{b,i} h(t - t'_{b,i}) \\ u'_{c,0} + \sum_{i=1}^{n_c} \Delta u'_{c,i} h(t - t'_{c,i}) \end{bmatrix} \tag{7.27}$$

is the three-phase absolute incumbent pulse pattern, and

$$\tilde{\boldsymbol{u}}'_{abc}(t, \lambda_{p,i}) = \begin{bmatrix} \tilde{u}'_a(t, \lambda_{a,i}) \\ \tilde{u}'_b(t, \lambda_{b,i}) \\ \tilde{u}'_c(t, \lambda_{c,i}) \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{n_a} \frac{\Delta u'_{a,i}}{\Delta u_{a,i}} \lambda_{a,i} \delta(t - t'_{a,i}) \\ \sum_{i=1}^{n_b} \frac{\Delta u'_{b,i}}{\Delta u_{b,i}} \lambda_{b,i} \delta(t - t'_{b,i}) \\ \sum_{i=1}^{n_c} \frac{\Delta u'_{c,i}}{\Delta u_{c,i}} \lambda_{c,i} \delta(t - t'_{c,i}) \end{bmatrix} \tag{7.28}$$

is the three-phase absolute small-signal input. Here, n_p denotes the number of switching transitions in phase p that fall within the prediction horizon T_p. Note that the incumbent switching times are defined according to the current time instant $t_0 = 0$. The total number of switching transitions are denoted with $n_{\text{sw}} = n_a + n_b + n_c$.

7.4 Small-Signal Controller with Integrated Balancing of the Neutral-Point Potential

This section demonstrates how balancing of the neutral-point potential is integrated in the formulation of the (standard) small-signal controller. First, Section 7.4.1 derives the internal dynamic model, which now also models the neutral-point potential v_n, of the controller. Then, in Section 7.4.2, an additional term, that penalizes the deviation of the neutral-point potential v_n from its steady-state trajectory v_n^*, is added to the objective function of the small-signal controller described in Section 5.5.2. Finally, the optimization problem is presented in Section 7.4.3.

7.4.1 Internal Dynamic Model

The neutral-point potential v_n is now integrated into the internal dynamic model of the standard controller in Section 5.5.1 in order to predict the (complete) state vector $\boldsymbol{x}_{\text{np}}$ over a prediction horizon T_p as a function of the modified pulse pattern $\boldsymbol{u}_{\text{mod},abc}$. However, due to the switched behaviour of the state matrix $\boldsymbol{F}_{\text{np}}(\boldsymbol{u}'_{\text{mod},abc})$, it is not possible to (easily) represent the predicted state vector $\boldsymbol{x}_{\text{np}}$ using a single set of equations [see (7.8), which describes the steady-state trajectory $\boldsymbol{x}^*_{\text{np}}$]. Furthermore, since the absolute modified pulse pattern $\boldsymbol{u}'_{\text{mod},abc}$, and therefore the state matrix $\boldsymbol{F}_{\text{np}}(\boldsymbol{u}'_{\text{mod},abc})$ as well, is a function of the impulse strengths $\lambda_{p,i}$, the solution to the differential equations will not have the well-known forms of (7.8). In order to derive an internal dynamic model that is practically useful, the neutral-point potential v_n is assumed to be constant when its effect on the (load) state vector $\boldsymbol{x}$ is considered. This assumption reduces the state-space representation that describes the state vector to

$$\frac{\mathrm{d}\boldsymbol{x}(t)}{\mathrm{d}t} = \boldsymbol{F}\boldsymbol{x}(t) + \boldsymbol{G}\boldsymbol{u}_{abc}(t) + \boldsymbol{G}_n(v_{n,0})\boldsymbol{u}'_{abc}(t) + \boldsymbol{P}\boldsymbol{v}_g(t), \tag{7.29}$$

where $v_{n,0}$ is the initial neutral-point potential (at time $t_0 = 0$), and $\boldsymbol{G}_n(v_{n,0}) \in \mathbb{R}^{n_x \times n_u}$ is introduced as

$$\boldsymbol{G}_n = \frac{v_{n,0}}{L}\boldsymbol{K}.$$

Now, the neutral-point potential v_n is no longer a state variable when considering the (load) state vector $\boldsymbol{x}$. Note that the evolution of the neutral-point potential is still

$$\frac{\mathrm{d}v_n(t)}{\mathrm{d}t} = \frac{1}{2C_d}\left(\boldsymbol{x}^{\mathrm{T}}(t)\boldsymbol{T}^{\mathrm{T}}\boldsymbol{u}'_{abc}(t)\right), \tag{7.30}$$

and fully takes the (reduced) load state vector $\boldsymbol{x}$ into account. This assumption, in a sense, decouples the converter system into two models: one that describes the load state vector $\boldsymbol{x}$ and another that describes the neutral-point potential v_n.

Next, the model that describes the predicted state vector $\boldsymbol{x}$ is derived. Thereafter, by using the model that describes the state vector $\boldsymbol{x}$, the model of the neutral-point potential v_n is derived. Before continuing, recall from Section 5.5.1 that the small-signal error is defined as

$$\tilde{\boldsymbol{x}}(t) = \boldsymbol{x}(t) - \boldsymbol{x}^*(t), \tag{7.31}$$

where (5.28) is repeated here for convenience. Similarly, the *small-signal neutral-point error* is defined as

$$\tilde{v}_n(t) = v_n(t) - v_n^*(t). \tag{7.32}$$

The Small-Signal Error

By integrating (7.29) with the modified pulse pattern $\boldsymbol{u}_{abc,\text{mod}}$ as the input, the state vector can be predicted at time $t \in [0, T_p]$ as

$$\begin{aligned}\boldsymbol{x}(t, \lambda_{p,i}) = \boldsymbol{e}^{\boldsymbol{F}t}\boldsymbol{x}_0 &+ \int_0^t \boldsymbol{e}^{\boldsymbol{F}(t-\tau)}\boldsymbol{G}\boldsymbol{u}_{abc,\text{mod}}(\tau, \lambda_{p,i})\,\mathrm{d}\tau + \int_0^t \boldsymbol{e}^{\boldsymbol{F}(t-\tau)}\boldsymbol{G}_n(v_{n,0})\boldsymbol{u}'_{abc,\text{mod}}(\tau, \lambda_{p,i})\,\mathrm{d}\tau \\ &+ \int_0^t \boldsymbol{e}^{\boldsymbol{F}(t-\tau)}\boldsymbol{P}\boldsymbol{v}_g(\tau)\,\mathrm{d}\tau,\end{aligned} \tag{7.33}$$

where $\boldsymbol{x}_0$ is the initial (sampled) state at time $t_0 = 0$. Similarly, the (reduced) steady-state trajectory follows from the nominal pulse pattern $\boldsymbol{u}^*_{abc}$ as

$$\begin{aligned}\boldsymbol{x}^*(t) = \boldsymbol{e}^{\boldsymbol{F}t}\boldsymbol{x}^*_0 + \int_0^t \boldsymbol{e}^{\boldsymbol{F}(t-\tau)}\boldsymbol{G}\boldsymbol{u}^*_{abc}(\tau)\,\mathrm{d}\tau + \int_0^t \boldsymbol{e}^{\boldsymbol{F}(t-\tau)}\boldsymbol{G}_n(v^*_{n,0})\boldsymbol{u}'^*_{abc}(\tau)\,\mathrm{d}\tau \\ + \int_0^t \boldsymbol{e}^{\boldsymbol{F}(t-\tau)}\boldsymbol{P}\boldsymbol{v}_g(\tau)\,\mathrm{d}\tau,\end{aligned} \tag{7.34}$$

where $\boldsymbol{x}^*_0$ and $v^*_{n,0}$ are the initial optimal state and the initial optimal neutral-point potential, respectively.[1] After inserting (7.33) and (7.34) into (7.31), and using the fact that $\boldsymbol{u}^*_{abc} = \boldsymbol{u}_{abc}$ and $\boldsymbol{u}'^*_{abc} = \boldsymbol{u}'_{abc}$ are assumed for the standard small-signal controller, the small-signal error follows as

$$\begin{aligned}\tilde{\boldsymbol{x}}(t,\lambda_{p,i}) = \boldsymbol{e}^{\boldsymbol{F}t}\tilde{\boldsymbol{x}}_0 + \int_0^t \boldsymbol{e}^{\boldsymbol{F}(t-\tau)}\boldsymbol{G}_n(\tilde{v}_{n,0})\boldsymbol{u}'_{abc}(\tau)\,\mathrm{d}\tau \\ + \int_0^t \boldsymbol{e}^{\boldsymbol{F}(t-\tau)}\boldsymbol{G}\tilde{\boldsymbol{u}}_{abc}(\tau,\lambda_{p,i})\,\mathrm{d}\tau + \int_0^t \boldsymbol{e}^{\boldsymbol{F}(t-\tau)}\boldsymbol{G}_n(v_{n,0})\tilde{\boldsymbol{u}}'_{abc}(\tau,\lambda_{p,i})\,\mathrm{d}\tau.\end{aligned} \tag{7.35}$$

Next, the small-signal error $\tilde{\boldsymbol{x}}$ is written in a compact vector form. By decomposing the small-signal input $\tilde{\boldsymbol{u}}_{abc}$ and the absolute small-signal input $\tilde{\boldsymbol{u}}'_{abc}$ into each phase, (7.35) becomes

$$\begin{aligned}\tilde{\boldsymbol{x}}(t,\lambda_{p,i}) = \boldsymbol{e}^{\boldsymbol{F}t}\tilde{\boldsymbol{x}}_0 + \boldsymbol{\phi}(t) + \int_0^t \boldsymbol{e}^{\boldsymbol{F}(t-\tau)}\boldsymbol{G}\left(\begin{bmatrix}1\\0\\0\end{bmatrix}\sum_{i=1}^{n_a}\lambda_{a,i}\delta(\tau - t'_{a,i}) + \begin{bmatrix}0\\1\\0\end{bmatrix}\sum_{i=1}^{n_b}\lambda_{b,i}\delta(\tau - t'_{b,i})\right. \\ \left. + \begin{bmatrix}0\\0\\1\end{bmatrix}\sum_{i=1}^{n_c}\lambda_{c,i}\delta(\tau - t'_{c,i})\right)\mathrm{d}\tau + \int_0^t \boldsymbol{e}^{\boldsymbol{F}(t-\tau)}\boldsymbol{G}_n(v_{n,0})\left(\begin{bmatrix}1\\0\\0\end{bmatrix}\sum_{i=1}^{n_a}\frac{\Delta u'_{a,i}}{\Delta u_{a,i}}\lambda_{a,i}\delta(\tau - t'_{a,i})\right. \\ \left. + \begin{bmatrix}0\\1\\0\end{bmatrix}\sum_{i=1}^{n_b}\frac{\Delta u'_{b,i}}{\Delta u_{b,i}}\lambda_{b,i}\delta(\tau - t'_{b,i})\begin{bmatrix}0\\0\\1\end{bmatrix}\sum_{i=1}^{n_c}\frac{\Delta u'_{c,i}}{\Delta u_{c,i}}\lambda_{c,i}\delta(\tau - t'_{c,i})\right)\mathrm{d}\tau\end{aligned}$$

where $\boldsymbol{\phi} \in \mathbb{R}^{n_x}$ is introduced as

$$\boldsymbol{\phi}(t) = \int_0^t \boldsymbol{e}^{\boldsymbol{F}(t-\tau)}\boldsymbol{G}_n(\tilde{v}_{n,0})\boldsymbol{u}'_{abc}(\tau)\,\mathrm{d}\tau.$$

By introducing $\boldsymbol{G}_{a,i} = \left(\boldsymbol{G} + \frac{\Delta u'_{a,i}}{\Delta u_{a,i}}\boldsymbol{G}_n(v_{n,0})\right)[1\ 0\ 0]^{\mathrm{T}}$, $\boldsymbol{G}_{b,i} = \left(\boldsymbol{G} + \frac{\Delta u'_{b,i}}{\Delta u_{b,i}}\boldsymbol{G}_n(v_{n,0})\right)[0\ 1\ 0]^{\mathrm{T}}$, and $\boldsymbol{G}_{c,i} = \left(\boldsymbol{G} + \frac{\Delta u'_{c,i}}{\Delta u_{c,i}}\boldsymbol{G}_n(v_{n,0})\right)[0\ 0\ 1]^{\mathrm{T}}$, and using the well-known sifting property of the impulse function, the small-signal error becomes

$$\begin{aligned}\tilde{\boldsymbol{x}}(t,\lambda_{p,i}) = \boldsymbol{e}^{\boldsymbol{F}t}\tilde{\boldsymbol{x}}_0 + \boldsymbol{\phi}(t) + \left(\sum_{i=1}^{n_a}\boldsymbol{e}^{\boldsymbol{F}(t-t'_{a,i})}\boldsymbol{G}_{a,i}h(t - t'_{a,i})\lambda_{a,i}\right. \\ \left. + \sum_{i=1}^{n_b}\boldsymbol{e}^{\boldsymbol{F}(t-t'_{b,i})}\boldsymbol{G}_{b,i}h(t - t'_{b,i})\lambda_{b,i} + \sum_{i=1}^{n_c}\boldsymbol{e}^{\boldsymbol{F}(t-t'_{c,i})}\boldsymbol{G}_{c,i}h(t - t'_{c,i})\lambda_{c,i}\right).\end{aligned}$$

By using basic algebraic manipulations, the small-signal error can be compactly stated as

$$\tilde{\boldsymbol{x}}(t,\boldsymbol{\Lambda}) = \boldsymbol{e}^{\boldsymbol{F}t}\tilde{\boldsymbol{x}}_0 + \boldsymbol{\phi}(t) + \boldsymbol{\Phi}_n(t)\boldsymbol{\Lambda} \tag{7.36}$$

[1] Note that initial optimal values $\boldsymbol{x}^*_0$ and $v^*_{n,0}$ are calculated according to (7.15).

where $\boldsymbol{\Phi}_n \in \mathbb{R}^{n_x \times n_{\text{sw}}}$ is the input matrix,

$$\begin{aligned}\boldsymbol{\Phi}_n(t) = \Big[& \boldsymbol{e}^{\boldsymbol{F}(t-t'_{a,1})}\boldsymbol{G}_{a,1}h(t-t'_{a,1}) \ \cdots \ \boldsymbol{e}^{\boldsymbol{F}(t-t'_{a,n_a})}\boldsymbol{G}_{a,n_a}h(t-t'_{a,n_a}) \\ & \boldsymbol{e}^{\boldsymbol{F}(t-t'_{b,1})}\boldsymbol{G}_{b,1}h(t-t'_{b,1}) \ \cdots \ \boldsymbol{e}^{\boldsymbol{F}(t-t'_{b,n_b})}\boldsymbol{G}_{b,n_b}h(t-t'_{b,n_b}) \\ & \boldsymbol{e}^{\boldsymbol{F}(t-t'_{c,1})}\boldsymbol{G}_{c,1}h(t-t'_{c,1}) \ \cdots \ \boldsymbol{e}^{\boldsymbol{F}(t-t'_{c,n_c})}\boldsymbol{G}_{c,n_c}h(t-t'_{c,n_c}) \Big],\end{aligned} \tag{7.37}$$

and $\boldsymbol{\Lambda} \in \mathbb{R}^{n_{sw}}$ is the strength vector,

$$\boldsymbol{\Lambda} = \begin{bmatrix} \lambda_{a,1} & \cdots & \lambda_{a,n_a} & \lambda_{b,1} & \cdots & \lambda_{b,n_b} & \lambda_{c,1} & \cdots & \lambda_{c,n_c} \end{bmatrix}^{\mathrm{T}}. \tag{7.38}$$

The Small-Signal Neutral-Point Error

By integrating (7.30) with the modified pulse pattern $\boldsymbol{u}_{abc,\text{mod}}$ as the input, the neutral-point potential can be predicted at time $t \in [0, T_p]$ as

$$v_n(t, \lambda_{p,i}) = v_{n,0} + \frac{1}{2C_d}\int_0^t \boldsymbol{x}^{\mathrm{T}}(\tau, \lambda_{p,i})\boldsymbol{T}^{\mathrm{T}}\boldsymbol{u}'_{mod,abc}(\tau, \lambda_{p,i})\,\mathrm{d}\tau. \tag{7.39}$$

Similarly, the steady-state trajectory of the neutral-point potential follows from the nominal pulse pattern $\boldsymbol{u}^*_{abc}$ as

$$v^*_n(t) = v^*_{n,0} + \frac{1}{2C_d}\int_0^t \boldsymbol{x}^{*\mathrm{T}}(\tau)\boldsymbol{T}^{\mathrm{T}}\boldsymbol{u}'^*_{abc}(\tau)\,\mathrm{d}\tau. \tag{7.40}$$

From the definition of (7.32), and by inserting (7.39) and (7.40), the small-signal neutral-point error follows as

$$\begin{aligned}\tilde{v}_n(t, \lambda_{p,i}) = \tilde{v}_{n,0} + \frac{1}{2C_d}\int_0^t \Big(& \tilde{\boldsymbol{x}}^{\mathrm{T}}(\tau, \lambda_{p,i})\boldsymbol{T}^{\mathrm{T}}\tilde{\boldsymbol{u}}'_{abc}(\tau, \lambda_{p,i}) + \tilde{\boldsymbol{x}}^{\mathrm{T}}(\tau, \lambda_{p,i})\boldsymbol{T}^{\mathrm{T}}\boldsymbol{u}'_{abc}(\tau) \\ & + \boldsymbol{x}^{*\mathrm{T}}(\tau)\boldsymbol{T}^{\mathrm{T}}\tilde{\boldsymbol{u}}'_{abc}(\tau, \lambda_{p,i}) \Big)\,\mathrm{d}\tau.\end{aligned} \tag{7.41}$$

By inserting the small-signal error of (7.36) into (7.41), and solving the integral, leads to the compact expression

$$\tilde{v}_n(t, \boldsymbol{\Lambda}) = \tilde{v}_{n,0} + \theta(t) + (\boldsymbol{m}_1(t) + \boldsymbol{m}_2(t) + \boldsymbol{m}_3(t))^{\mathrm{T}}\boldsymbol{\Lambda} + \boldsymbol{\Lambda}^{\mathrm{T}}\boldsymbol{M}(t)\boldsymbol{\Lambda} \tag{7.42}$$

for the small-signal neutral-point error, where $\theta \in \mathbb{R}$, $\boldsymbol{m}_1 \in \mathbb{R}^{n_{\text{sw}}}$, $\boldsymbol{m}_2 \in \mathbb{R}^{n_{\text{sw}}}$, $\boldsymbol{m}_3 \in \mathbb{R}^{n_{\text{sw}}}$, and $\boldsymbol{M} \in \mathbb{R}^{n_{\text{sw}} \times n_{\text{sw}}}$ are defined as

$$\boldsymbol{m}_1^{\mathrm{T}}(t)\boldsymbol{\Lambda} + \boldsymbol{\Lambda}^{\mathrm{T}}\boldsymbol{M}(t)\boldsymbol{\Lambda} = \frac{1}{2C_d}\int_0^t \tilde{\boldsymbol{x}}^{\mathrm{T}}(\tau, \lambda_{p,i})\boldsymbol{T}^{\mathrm{T}}\tilde{\boldsymbol{u}}'_{abc}(\tau, \lambda_{p,i})\,\mathrm{d}\tau \tag{7.43}$$

$$\theta(t) + \boldsymbol{m}_2^{\mathrm{T}}(t)\boldsymbol{\Lambda} = \frac{1}{2C_d}\int_0^t \tilde{\boldsymbol{x}}^{\mathrm{T}}(\tau, \lambda_{p,i})\boldsymbol{T}^{\mathrm{T}}\boldsymbol{u}'_{abc}(\tau)\,\mathrm{d}\tau \tag{7.44}$$

$$\boldsymbol{m}_3^{\mathrm{T}}(t)\boldsymbol{\Lambda} = \frac{1}{2C_d}\int_0^t \boldsymbol{x}^{*\mathrm{T}}(\tau)\boldsymbol{T}^{\mathrm{T}}\tilde{\boldsymbol{u}}'_{abc}(\tau, \lambda_{p,i})\,\mathrm{d}\tau. \tag{7.45}$$

For the sake of brevity, the derivations of (7.43)–(7.45) are moved to Appendix F.

Observe that the small-signal neutral-point error $\tilde{v}_n$ is quadratic in the strength vector $\boldsymbol{\Lambda}$. Unfortunately, this means that the optimization problem formulated in Section 7.4.3 will not be a QP (with linear constraints) if (7.42) is used. Instead, the optimization problem will either be a nonlinear fourth-order program (if $\tilde{v}_n$ is weighted with a 2-norm) or a QP with nonconvex quadratic

constraints (if a 1-norm weighting function is used).[2] In order to guarantee a QP (with linear constraints) in Section 7.4.3, the quadratic term is ignored, reducing the small-signal neutral-point error of (7.42) to

$$\tilde{v}_n(t, \boldsymbol{\Lambda}) = \tilde{v}_{n,0} + \theta(t) + \boldsymbol{m}^{\mathrm{T}}(t)\boldsymbol{\Lambda}, \tag{7.46}$$

where $\boldsymbol{m} = \boldsymbol{m}_1 + \boldsymbol{m}_2 + \boldsymbol{m}_3$. By ignoring $\boldsymbol{M}$, second-order behaviour is not taken into account. Specifically, the way in which the small-signal input $\tilde{\boldsymbol{u}}'_{abc}$ interacts with the absolute small-signal input $\tilde{\boldsymbol{u}}'_{abc}$ is not modelled (that is, how corrections to the incumbent pulse pattern interact with corrections to the absolute incumbent pulse pattern). Fortunately, the remaining terms still model useful and important behaviour. Specifically

- $\boldsymbol{m}_1$ models how the absolute small-signal input $\tilde{\boldsymbol{u}}'_{abc}$ interacts with the *unforced* small-signal error $\boldsymbol{x}$ (that is, when $\tilde{\boldsymbol{u}}_{abc}$ is zero),
- $\boldsymbol{m}_2$ and θ model how the absolute incumbent pulse pattern $\boldsymbol{u}'_{abc}$ interacts with the small-signal error $\boldsymbol{x}$ (in other words, how the corrections to the converter current interact with the absolute incumbent pulse pattern), and
- $\boldsymbol{m}_3$ models how the absolute small-signal input $\tilde{\boldsymbol{u}}'_{abc}$ interacts with the steady-state trajectory $\boldsymbol{x}^*$.

7.4.2 Objective Function

The control objectives are to minimize the small-signal error across the prediction horizon, to penalize modifications to the incumbent pulse pattern (that is, the control effort), and to minimize the small-signal neutral-point error. Mapping these control objectives into an objective function results in

$$J(\boldsymbol{\Lambda}) = J_1(\boldsymbol{\Lambda}) + J_2(\boldsymbol{\Lambda}) + J_{\mathrm{np}}(\boldsymbol{\Lambda}) \tag{7.47}$$

where J_1 and J_2 are defined as

$$J_1(\boldsymbol{\Lambda}) = \int_0^{T_p} \tfrac{1}{2}\|\tilde{\boldsymbol{x}}(t, \boldsymbol{\Lambda})\|^2_{\boldsymbol{Q}} \,\mathrm{d}t$$
$$J_2(\boldsymbol{\Lambda}) = \tfrac{1}{2}\|\boldsymbol{\Lambda}\|^2_{\boldsymbol{R}},$$

respectively. Before choosing a suitable expression for the term J_{np}, note that the terms J_1 and J_2 follow from the objective function of the small-signal controller *without* integrated balancing of the neutral-point potential [see (5.35), the original objective function]. Furthermore, note that the standard model of the small-signal controller that takes the initial neutral-point potential $v_{n,0}$ into account [which is used in this chapter, and shown in (7.36)] and the advanced model of the small-signal controller without the neutral-point potential [as shown in (5.31)] have a near identical algebraic structure. This means that the derivations of the terms J_1 and J_2 will be near identical to those detailed in Section 5.5.2. For the sake of brevity, and to avoid repetition, these terms are not expanded again; only the (final) standard quadratic form is given here as

$$J_1(\boldsymbol{\Lambda}) + J_2(\boldsymbol{\Lambda}) = \tfrac{1}{2}\boldsymbol{\Lambda}^{\mathrm{T}}\boldsymbol{H}\boldsymbol{\Lambda} + \boldsymbol{c}^{\mathrm{T}}\boldsymbol{\Lambda}. \tag{7.48}$$

Ideally, the small-signal neutral-point error $\tilde{v}_n$ should be minimized across the prediction horizon according to

$$J_{\mathrm{np}}(\boldsymbol{\Lambda}) = \int_0^{T_p} \tfrac{1}{2} q_{\mathrm{np}}(\tilde{v}_n(t, \boldsymbol{\Lambda}))^2 \,\mathrm{d}t,$$

[2] Although the definiteness of $\boldsymbol{M}$ is not established algebraically, numerical simulations show it is indefinite.

where $q_{\text{np}} \in \mathbb{R}$ is the nonnegative penality associated with the small-signal neutral-point error $\tilde{v}_n$. Using (7.46), the integrand can be expanded to

$$J_{\text{np}}(\boldsymbol{\Lambda}) = \tfrac{1}{2} q_{\text{np}} \int_0^{T_p} \left(\boldsymbol{\Lambda}^{\mathrm{T}} \boldsymbol{m}(t) \boldsymbol{m}^{\mathrm{T}}(t) \boldsymbol{\Lambda} + 2 \boldsymbol{m}^{\mathrm{T}}(t) \left(\tilde{v}_{n,0} + \theta(t) \right) \boldsymbol{\Lambda} + \left(\tilde{v}_{n,0} + \theta(t) \right)^2 \right) \mathrm{d}t.$$

However, θ and $\boldsymbol{m}$ both contain a moderate number of matrix exponentials that are required to be integrated. In order to reduce the computational burden associated with the balancing of the neutral-point potential, the small-signal neutral-point error $\tilde{v}_n$ is only penalized at the end of the prediction horizon according to

$$J_{\text{np}}(\boldsymbol{\Lambda}) = \tfrac{1}{2} q_{\text{np}} \left(\tilde{v}_n(T_p, \boldsymbol{\Lambda}) \right)^2 . \tag{7.49}$$

By inserting (7.46) into (7.49), it follows that

$$J_{\text{np}}(\boldsymbol{\Lambda}) = \tfrac{1}{2} q_{\text{np}} \boldsymbol{\Lambda}^{\mathrm{T}} \boldsymbol{H}_{\text{np}} \boldsymbol{\Lambda} + q_{\text{np}} \boldsymbol{c}_{\text{np}}, \tag{7.50}$$

where terms that are not a function of the strength vector $\boldsymbol{\Lambda}$ have been ignored. The Hessian $\boldsymbol{H}_{\text{np}} \in \mathbb{R}^{n_{\text{sw}} \times n_{\text{sw}}}$ and vector $\boldsymbol{c}_{\text{np}} \in \mathbb{R}^{n_{\text{sw}}}$ associated with small-signal neutral-point error are

$$\boldsymbol{H}_{\text{np}} = \boldsymbol{m}(T_p) \boldsymbol{m}^{\mathrm{T}}(T_p) \tag{7.51}$$

$$\boldsymbol{c}_{\text{np}} = \boldsymbol{m}(T_p) \left(\tilde{v}_{n,0} + \theta(T_p) \right), \tag{7.52}$$

respectively.

Adding (7.50) to (7.48) yields the quadratic objective function

$$J(\boldsymbol{\Lambda}) = \tfrac{1}{2} \boldsymbol{\Lambda}^{\mathrm{T}} \left(\boldsymbol{H} + q_{\text{np}} \boldsymbol{H}_{\text{np}} \right) \boldsymbol{\Lambda} + \left(\boldsymbol{c} + q_{\text{np}} \boldsymbol{c}_{\text{np}} \right)^{\mathrm{T}} \boldsymbol{\Lambda}. \tag{7.53}$$

To summarize, the quadratic objective function contains two parts. The first part relates to the original objectives of the small-signal controller as defined in Section 5.5.2, which is realized by $\boldsymbol{H}$ and $\boldsymbol{c}$. The second part refers to the integrated balancing of the neutral-point potential, and is realized by $\boldsymbol{H}_{\text{np}}$ and $\boldsymbol{c}_{\text{np}}$.

7.4.3 Optimization

With the addition of balancing of the neutral-point potential, the QP of (5.54) now becomes

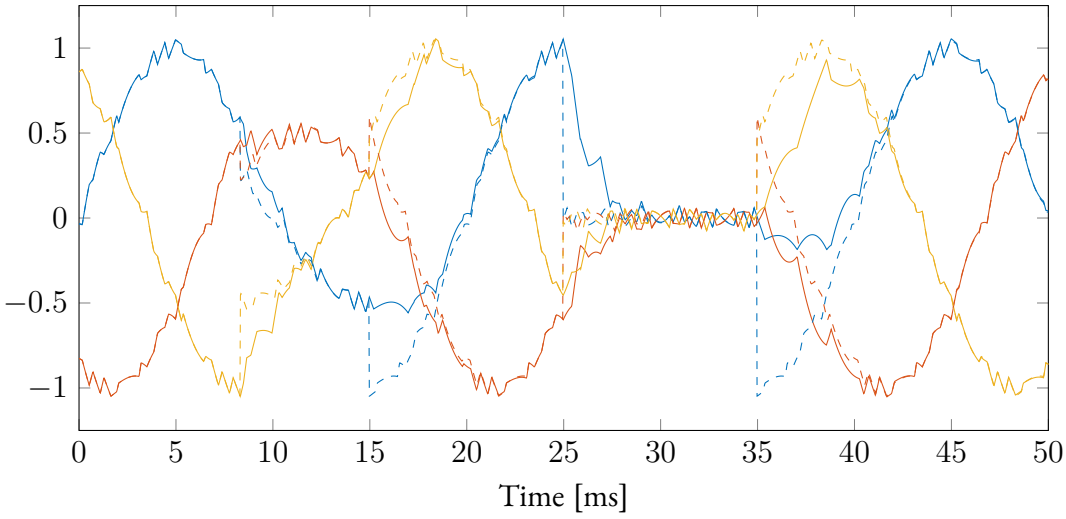
$$\begin{aligned} \boldsymbol{\Lambda}_{\text{opt}} = \arg\min_{\boldsymbol{\Lambda}} \quad & \tfrac{1}{2} \boldsymbol{\Lambda}^{\mathrm{T}} \left(\boldsymbol{H} + q_{\text{np}} \boldsymbol{H}_{\text{np}} \right) \boldsymbol{\Lambda} + \left(\boldsymbol{c} + q_{\text{np}} \boldsymbol{c}_{\text{np}} \right)^{\mathrm{T}} \boldsymbol{\Lambda} \\ & \text{subject to } \boldsymbol{A} \boldsymbol{\Lambda} \leq \boldsymbol{b}, \end{aligned}$$

where the constraints are defined in Section 5.5.3.

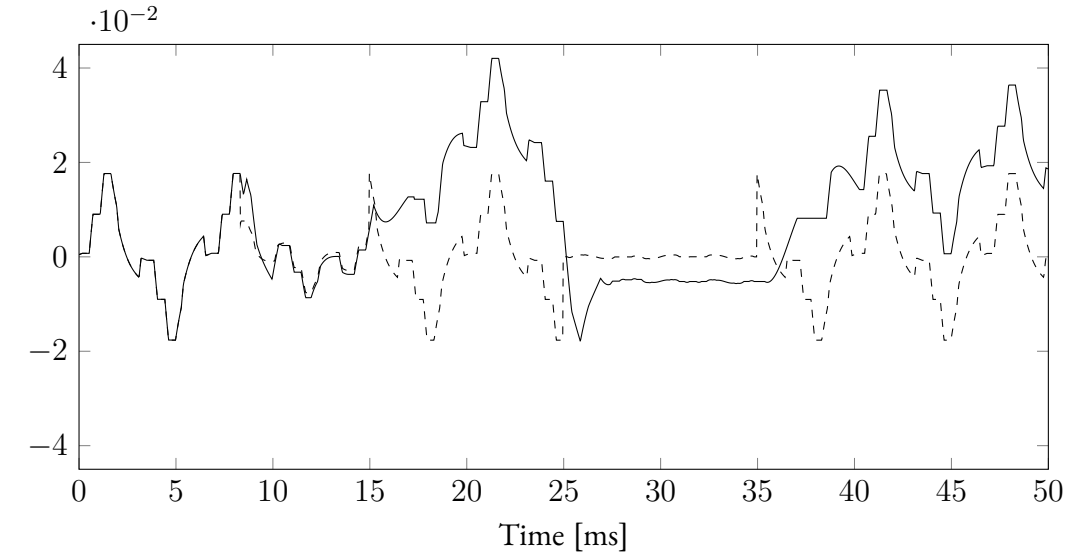
7.5 Performance Evaluation

This section evaluates the efficacy of the standard small-signal controller with integrated balancing of the neutral-point potential via simulations. First, the performance is evaluated during the dynamic operation of the converter system. Then, performance is evaluated at the case where traditional methods fail: when the converter current and voltage are 90 degrees out of phase (zero power factor at the converter terminals).

As a case study, a first-order converter system is used (the parameters are given in Section 7.1). The device switching frequency is set to $f_{\text{sw}} = 250\,\text{Hz}$ (a pulse number of $d = 5$). The prediction horizon and sampling interval of the controller are set to $T_p = 2\,\text{ms}$ and $T_s = 25\,\mu\text{s}$, respectively. All state variables are penalized equally with $\boldsymbol{Q} = \mathbf{I}_2$ and all switching transition modifications are penalized equally with $\boldsymbol{R} = \mathbf{I}_{n_{\text{sw}}}$.



(a) Converter currents

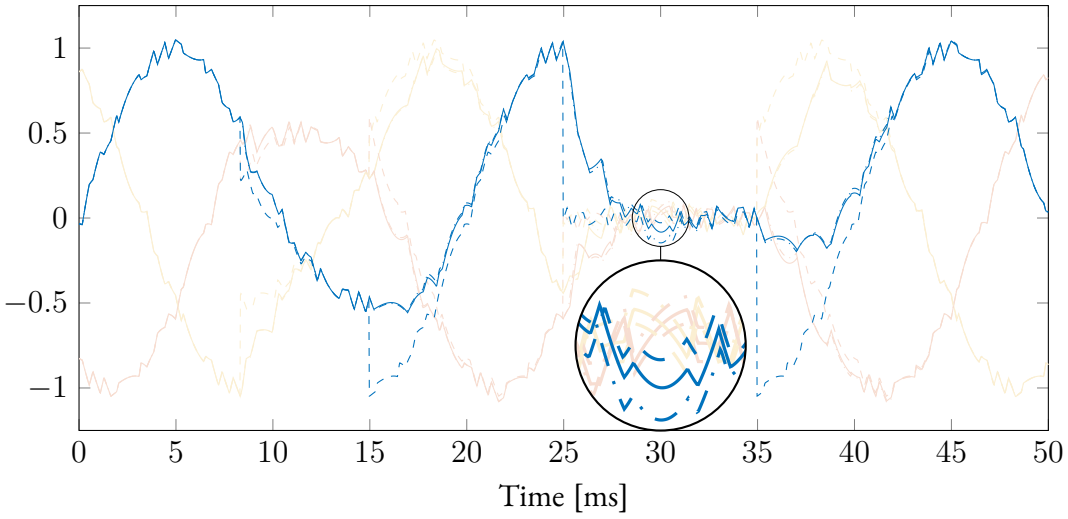


(b) Neutral-point potential

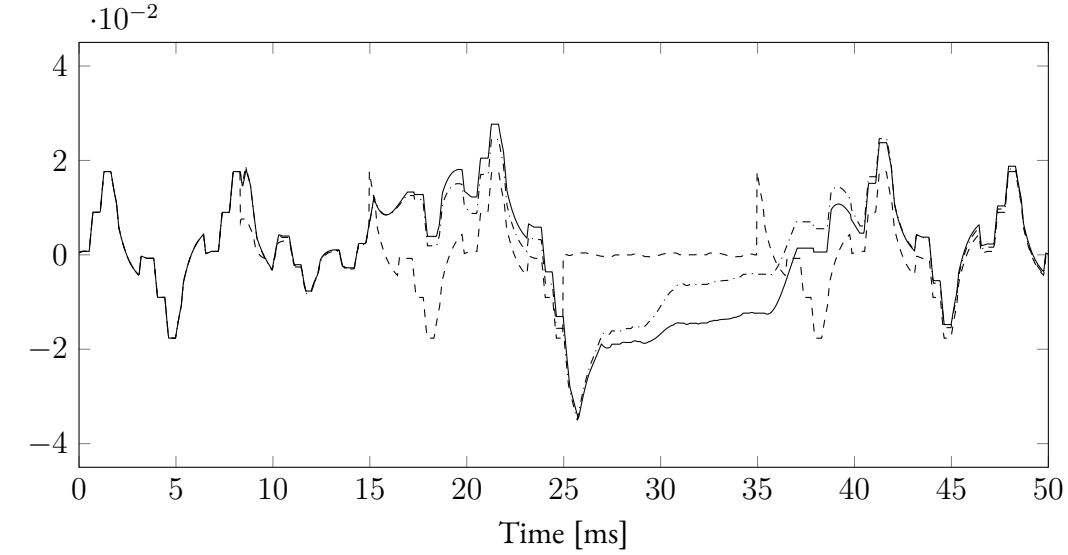
Figure 7.4: The converter states (in pu) during multiple reference steps without balancing of the neutral-point potential. References are indicated with dashed lines.

7.5.1 During Transients

Consider the case where the converter system is operating at unity power factor at the grid voltage source (the reactive power reference Q^* is set to zero). In Figure 7.4, the controller is given four power reference steps. The weight on the neutral-point potential balancing q_{np} is set to zero (and thus deactivated). Note that the internal dynamic model still takes the effect of the initial neutral-point potential $v_{n,0}$ into account. Initially, the power reference is $P^* = 1\,\text{pu}$ and the converter system is in the steady state. Then, at 8.35 ms, the power reference is stepped to $P^* = 0.5\,\text{pu}$. At 15 ms, the power reference is stepped back to $P^* = 1\,\text{pu}$. The power reference is then stepped to $P^* = 0\,\text{pu}$ and back to $P^* = 1\,\text{pu}$ at 25 ms and 35 ms, respectively. It can be seen that the small-signal controller exerts no effort in balancing the neutral-point potential v_n. Moreover, it can be observed that the natural balancing mechanisms are weak; once the converter is operating at rated power, the average value of the neutral-point potential v_n does not diminish. Typically, pulse patterns with high modulation indices do not possess strong natural balancing properties. Note



(a) Converter currents

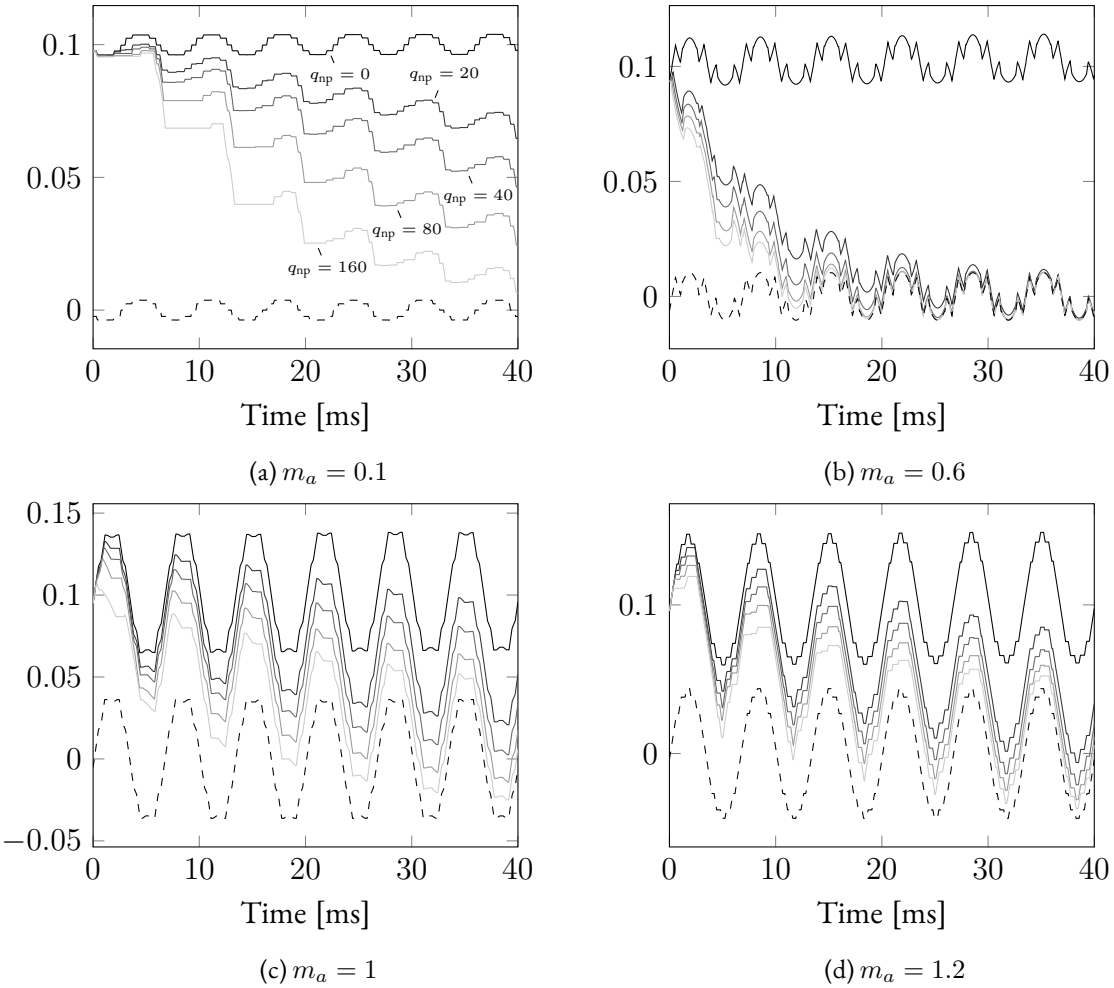


(b) Neutral-point potential

Figure 7.5: The converter states (in pu) during multiple reference steps with neutral-point potential balancing. References are indicated with dashed lines. The responses of the weights $q_{\text{np}} = 40$ and $q_{\text{np}} = 160$ are indicated with the solid and dash-dotted lines, respectively.

that when the converter system is operating at zero power, the converter current is nearly zero, and therefore the neutral-point potential v_n is weakly affected.

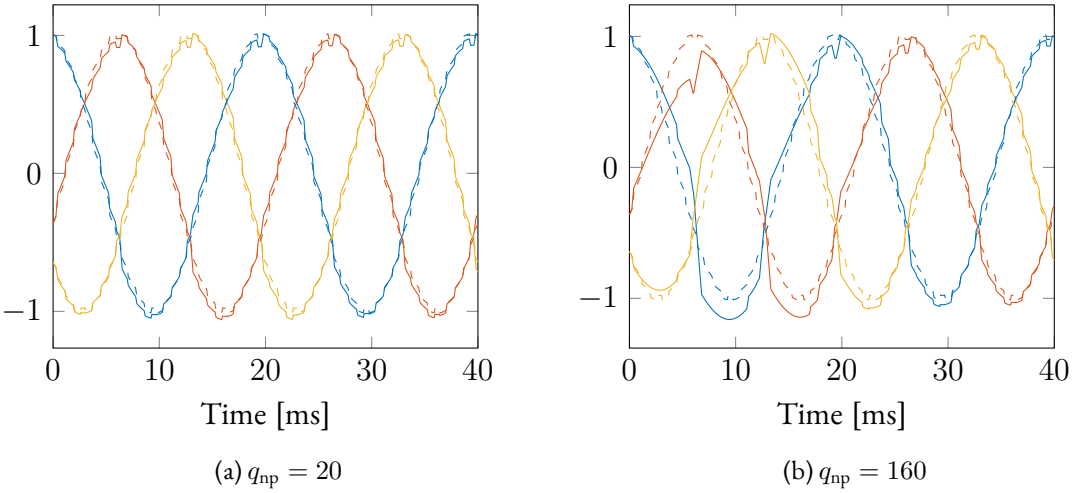
In Figure 7.5, the same reference steps as Figure 7.4 are given, but with weights $q_{\text{np}} = 40$ and $q_{\text{np}} = 160$ on the neutral-point potential balancing. Note that the former weight is small, since the Hessian $\boldsymbol{H}_{\text{np}}$ and vector $\boldsymbol{c}_{\text{np}}$ that refer to the balancing of the neutral-point potential are significantly smaller than the Hessian $\boldsymbol{H}$ and vector $\boldsymbol{c}$ that relate to the original control problem. For clarity, only the phase a converter current is clearly shown. As seen, the controller now balances the neutral-point potential v_n. The balancing is most notable when the converter system is operating at rated power; note the rate at which the neutral-point potential v_n balances in Figure 7.5 between 15 ms and 25 ms, and between 35 ms and 50 ms, when compared to Figure 7.4. Furthermore, it can be seen that the controller increases the ripple of the converter current in order to obtain control over the neutral-point potential; this is most notable at zero power, where it can be seen that a weight of $q_{\text{np}} = 160$ results in a larger current ripple when compared to a weight of $q_{\text{np}} = 40$.



(a) $m_a = 0.1$ (b) $m_a = 0.6$

(c) $m_a = 1$ (d) $m_a = 1.2$

Figure 7.6: The neutral-point potential (in pu) under zero power factor.



(a) $q_{np} = 20$ (b) $q_{np} = 160$

Figure 7.7: Converter current (in pu) when balancing the neutral-point potential under zero power factor.

7.5.2 Zero Power Factor at Converter Terminals

In this section, the balancing method is investigated for the case where the phase between the converter voltage and current is 90 degrees; traditional balancing methods that rely on common-mode voltage manipulation fail under this operating condition. Modulation indices m_a from 0.1 to 1.2 in

steps of 0.1 are considered in order to evaluate multiple pulse patterns. Different weighting factors are also considered. The grid voltage is adjusted so that the magnitude of the converter current is 1 pu. Initially, the converter is in the steady state, when a large offset of 0.1 pu is added to the neutral-point potential v_n. Simulations indicate that the controller is able to balance the neutral-point potential v_n at all the modulation indices that are considered. In Figure 7.6, the neutral-point potential v_n for the modulation indices $m_a \in \{0.1, 0.6, 1, 1.2\}$ is shown. As seen, without balancing, the average value of neutral-point potential v_n does not decay noticeably. An extremely small weight of $q_{\text{np}} = 20$ is already sufficient for the controller to balance the neutral-point potential v_n. Furthermore, it is clear that having a larger weight q_{np} results in faster balancing.

In order to see one of the mechanisms that the controller uses to obtain control over the neutral-point potential v_n under zero power factor at the converter terminals, consider the converter current shown in Figure 7.7 for a modulation index of $m_a = 0.1$. As seen, a larger weight q_{np} results in more aggressive modifications to the converter current; the controller does not rely on common-mode injection. Also note in Figure 7.7b that the modifications to the converter current diminish as the neutral-point potential v_n becomes balanced.

7.6 Summary

In this chapter, balancing of the neutral-point potential was integrated in the (standard) small-signal controller. Owing to the introduction of the so-called absolute pulse pattern u', the absolute value of a pulse pattern $|u|$ (and modifications to it) could easily be expressed and manipulated. A model that describes the neutral-point potential v_n was derived. The quadratic term of the model was ignored in order to ensure a linear model, resulting in the underlying optimization problem to be a QP. Fortunately, the linear model still captured useful and important behaviour.

Simulation results demonstrated that the controller balances the neutral-point potential v_n during the dynamic operation of a converter system. Importantly, the controller managed to balance the neutral-point potential v_n when operating at zero power factor at the converter terminals (that is, when the converter voltage and current were 90 degrees out of phase).

Chapter 8

Implementation of Standard Controller

This chapter presents an efficient implementation of the standard small-signal controller from Section 5.5.7. Specific attention is given to formulating and solving the QP, which is the part of the control algorithm with the highest computational burden. Other aspects of the control algorithm are not discussed. Specifically, this chapter explains how to calculate the Hessian and vector of the quadratic function, determine the stepsize of the gradient projection method, and implement the projection operator. The remainder of this chapter (briefly) discusses the implementation of the control algorithm on a field-programmable gate array. The grid-connected converter from Section 3.3.2 is considered as the case study. This chapter concludes with a verification of the implemented control algorithm operating in real-time.

Chapter Contents

8.1 Efficient Calculation of the Hessian and Vector

The first step regarding the QP is to calculate the Hessian $\boldsymbol{H} = \boldsymbol{V} + \boldsymbol{R}$ and vector $\boldsymbol{c}$ of the quadratic objective function. In this section, the structure of the Hessian $\boldsymbol{H}$ and vector $\boldsymbol{c}$ are analyzed and exploited in order to reduce the computational burden, with the aim of enabling an efficient implementation on a field-programmable gate array (FPGA). Section 8.1.1 gives a brief review of the calculations regarding the Hessian $\boldsymbol{H}$ and vector $\boldsymbol{c}$. Section 8.1.2 then shows how these calculations can be reduced. Note that the computations regarding the matrix exponentials are not discussed; it is assumed that the matrix exponentials are calculated offline and stored in lookup tables.

8.1.1 Review of the Objective Function

The (i', j')th component of $\boldsymbol{V}$ is defined in Section 5.5.2 as

$$\boldsymbol{V}_{(i',j')} = \boldsymbol{G}_{p_1}^{\mathrm{T}} \boldsymbol{e}^{\boldsymbol{F}^{\mathrm{T}}(t'_{p,ij} - t'_{p_1,i})} \boldsymbol{\Xi}(T_p - t'_{p,ij}) \boldsymbol{e}^{\boldsymbol{F}(t'_{p,ij} - t'_{p_2,j})} \boldsymbol{G}_{p_2}, \tag{8.1}$$

where $t'_{p,ij} = \max\{t'_{p_1,i}, t'_{p_2,j}\}$, and

$$\boldsymbol{\Xi}(\Delta T) = \int_0^{\Delta T} \boldsymbol{e}^{\boldsymbol{F}^{\mathrm{T}} t} \boldsymbol{Q} \boldsymbol{e}^{\boldsymbol{F} t} \, \mathrm{d}t.$$

The i'th index corresponds to phase $p_1 \in \{a, b, c\}$ and the ith switching transition in that phase. The j'th index is defined accordingly based on p_2. Refer to (5.42) on how to solve $\boldsymbol{\Xi}$. Furthermore, the i'th component of $\boldsymbol{c}$ is

$$\boldsymbol{c}_{i'}^{\mathrm{T}} = \tilde{\boldsymbol{x}}_0^{\mathrm{T}} \boldsymbol{e}^{\boldsymbol{F}^{\mathrm{T}} t'_{p,i}} \boldsymbol{\Xi}(T_p - t'_{p,i}) \boldsymbol{G}_p. \tag{8.2}$$

The matrix $\boldsymbol{R}$ is simply a diagonal matrix that penalizes the control effort.

8.1.2 Exploiting the Problem Structure

As a first step to reduce the computational burden, several aspects of the problem can be exploited. First, since $t'_{p,ij} = \max\{t'_{p_1,i}, t'_{p_2,j}\}$, it can be observed that either

$$\boldsymbol{e}^{\boldsymbol{F}^{\mathrm{T}}(t'_{p,ij} - t'_{p_1,i})} \quad \text{or} \quad \boldsymbol{e}^{\boldsymbol{F}(t'_{p,ij} - t'_{p_2,j})}$$

of (8.1) will be the identity matrix. This means that one of the matrix-vector products

$$\boldsymbol{G}_{p_1}^{\mathrm{T}} \boldsymbol{e}^{\boldsymbol{F}^{\mathrm{T}}(t'_{p,ij} - t'_{p_1,i})} \quad \text{or} \quad \boldsymbol{e}^{\boldsymbol{F}(t'_{p,ij} - t'_{p_2,j})} \boldsymbol{G}_{p_2},$$

where each product may require up to 36 multiplications, is not required to be calculated. Furthermore, note that $t'_{p_1,i} = t'_{p_2,j}$ holds when calculating the diagonal components of $\boldsymbol{V}$, leading to both

$$\boldsymbol{e}^{\boldsymbol{F}^{\mathrm{T}}(t'_{p,ij} - t'_{p_1,i})} \quad \text{and} \quad \boldsymbol{e}^{\boldsymbol{F}(t'_{p,ij} - t'_{p_2,j})}$$

being identity matrices. Thus, a diagonal component of $\boldsymbol{V}$ has the form

$$\boldsymbol{V}_{(i',j')} = \boldsymbol{G}_p^{\mathrm{T}} \boldsymbol{\Xi}(T_p - t'_{p,i}) \boldsymbol{G}_p, \tag{8.3}$$

where $p = p_1 = p_2$ and $t'_{p,i} = t'_{p_1,i} = t'_{p_2,j}$. Finally, note that the term

$$\boldsymbol{\Xi}(T_p - t'_{p,i}) \boldsymbol{G}_p$$

of (8.3) is present in (8.2). Thus, half of the terms of $\boldsymbol{c}_{i'}$ have already been calculated when the diagonal components of $\boldsymbol{V}$ are calculated.

Next, characteristics of the load are exploited in order to reduce the computational burden further. Recall from Section 5.1 that the state $\boldsymbol{F}$ and input $\boldsymbol{G}$ matrices characterizing the grid-connected converter are

$$\boldsymbol{F} = \begin{bmatrix} -\frac{R+R_C}{L}\mathbf{I}_2 & \frac{R_C}{L}\mathbf{I}_2 & -\frac{1}{L}\mathbf{I}_2 \\ \frac{R_C}{L_{gt}}\mathbf{I}_2 & -\frac{R_{gt}+R_C}{L_{gt}}\mathbf{I}_2 & \frac{1}{L_{gt}}\mathbf{I}_2 \\ \frac{1}{C}\mathbf{I}_2 & -\frac{1}{C}\mathbf{I}_2 & \mathbf{0}_{2\times 2} \end{bmatrix} \quad \text{and} \quad \boldsymbol{G} = \frac{V_d}{2L}\begin{bmatrix} \mathbf{I}_2 \\ \mathbf{0}_{2\times 2} \\ \mathbf{0}_{2\times 2} \end{bmatrix} \boldsymbol{K},$$

respectively, for a state vector that is defined as

$$\boldsymbol{x}(t) = \begin{bmatrix} i_\alpha(t) & i_\beta(t) & i_{g,\alpha}(t) & i_{g,\beta}(t) & v_{c,\alpha}(t) & v_{c,\beta}(t) \end{bmatrix}^{\mathrm{T}}.$$

Due to the α and β components of the state matrix $\boldsymbol{F}$ being decoupled, half of the components of the matrix exponential $\boldsymbol{e}^{\boldsymbol{F}t}$ are zero. More specifically, it is of the structure

$$\boldsymbol{e}^{\boldsymbol{F}t} = \begin{bmatrix} \mathrm{x} & 0 & \mathrm{x} & 0 & \mathrm{x} & 0 \\ 0 & \mathrm{x} & 0 & \mathrm{x} & 0 & \mathrm{x} \\ \mathrm{x} & 0 & \mathrm{x} & 0 & \mathrm{x} & 0 \\ 0 & \mathrm{x} & 0 & \mathrm{x} & 0 & \mathrm{x} \\ \mathrm{x} & 0 & \mathrm{x} & 0 & \mathrm{x} & 0 \\ 0 & \mathrm{x} & 0 & \mathrm{x} & 0 & \mathrm{x} \end{bmatrix},$$

where x denotes a nonzero element. Thus, the term

$$\tilde{\boldsymbol{x}}_0^{\mathrm{T}} \boldsymbol{e}^{\boldsymbol{F}^{\mathrm{T}} t'_{p,i}}$$

of (8.2) only requires 18 multiplications (instead of 36). Furthermore, recall that $\boldsymbol{G}_a = \boldsymbol{G}[1\ 0\ 0]^{\mathrm{T}}$, $\boldsymbol{G}_b = \boldsymbol{G}[0\ 1\ 0]^{\mathrm{T}}$, and $\boldsymbol{G}_c = \boldsymbol{G}[0\ 0\ 1]^{\mathrm{T}}$. Note that for any given phase $p \in \{a, b, c\}$ only the top two elements of $\boldsymbol{G}_p$ are nonzero. This results in matrix-vector multiplications in (8.1) of the forms

$$\boldsymbol{e}^{\boldsymbol{F}t}\boldsymbol{G}_p \quad \text{and} \quad \boldsymbol{\Xi}(\Delta T)\boldsymbol{G}_p$$

only requiring 6 multiplications (instead of 36).

8.2 Determining the Stepsize

Before the gradient projection method can be used to solve the QP, the (fixed) stepsize s of the method first needs to be determined. Section 8.2.1 shows how the (tight) Lipschitz constant L_c can be efficiently overestimated. Section 8.2.2 then demonstrates how the stepsize $s = \frac{1}{L_c}$ can be calculated using Newton's method.

8.2.1 Efficiently Overestimating the Lipschitz Constant

Recall from Section 2.3.2 that the optimal (fixed) stepsize for the gradient method that results in the fastest convergence is

$$s = \frac{2}{L_c + \mu},$$

where $L_c = \|\boldsymbol{H}\|_2$ (the tight Lipschitz constant) and $\mu = \frac{1}{\|\boldsymbol{H}^{-1}\|_2}$ (the tight convexity parameter) are the largest and smallest eigenvalues of the Hessian $\boldsymbol{H}$, respectively. First consider the Lipschitz

constant L_c, which requires the evaluation of the (induced) 2-norm of the Hessian $\boldsymbol{H}$. Evaluating the 2-norm of a matrix is computationally expensive, requiring an iterative method. Fortunately, a useful relation from [67, Section 2.3.3] states that the 2-norm of a matrix can be overestimated as

$$\|\boldsymbol{H}\|_2 \leq \sqrt{\|\boldsymbol{H}\|_1 \|\boldsymbol{H}\|_\infty},$$

and since the Hessian $\boldsymbol{H}$ is symmetric, the relation simplifies to

$$\|\boldsymbol{H}\|_2 \leq \|\boldsymbol{H}\|_1 = \|\boldsymbol{H}\|_\infty.$$

Evaluating the infinity-norm (or 1-norm) of a matrix is computationally very cheap, as it is simply

$$\|\boldsymbol{H}\|_\infty = \max_{1 \leq i \leq n_{\text{sw}}} \sum_{j=1}^{n_{\text{sw}}} \left| \boldsymbol{H}_{(i,j)} \right|, \tag{8.4}$$

which is the maximum of the absolute value of the matrix entries summed up in each row; no multiplications are required. Denote with $\bar{L}_c = \|\boldsymbol{H}\|_\infty$ the overestimated (tight) Lipschitz constant.

Unfortunately, an efficient approximation for the convexity parameter μ could not be found. If the convexity parameter μ is not available, the stepsize

$$s = \frac{1}{\bar{L}_c} \tag{8.5}$$

should be used (refer to Section 2.3.2 for more details). An overestimated (tight) Lipschitz constant $\bar{L}_c$ results in an underestimated stepsize s.

8.2.2 Calculating a Reciprocal

To calculate the stepsize $s = \frac{1}{\bar{L}_c}$, division needs to be implemented on the FPGA, since hardware description languages typically do not have a division primitive. A widely-used numerical method that can be used to realize division is Newton's method [68].

In a first step, the Lipschitz constant $\bar{L}_c$ is normalized to the interval $[0.5, 1]$ (this is common practice, and the reasoning for this will soon become apparent). The normalization can be easily achieved by using simple bit shifts, which are equivalent to dividing or multiplying by powers of two. Denote with $D = \bar{L}_c 2^m \in [0.5, 1]$ the normalized Lipschitz constant, where $m \in \mathbb{Z}$ is the number of bit shifts required to scale $\bar{L}_c$ to the interval $[0.5, 1]$. Then,

$$\frac{1}{D}$$

can be approximated with z by finding the root of

$$f(z) = \frac{1}{z} - D$$

by using Newton's method, where the kth iteration has the form [68]

$$z_{k+1} = z_k + z_k(1 - Dz_k). \tag{8.6}$$

Once Newton's method has terminated, the final iteration z_{opt} (which approximates $\frac{1}{D}$) is denormalized to find the stepsize as $s = z_{\text{opt}} 2^m$.

Note that in order for the method to converge, the initial solution z_0 must fall within the interval $(0, \frac{2}{D})$. Recall that Newton's method has quadratic convergence, meaning the error reduces

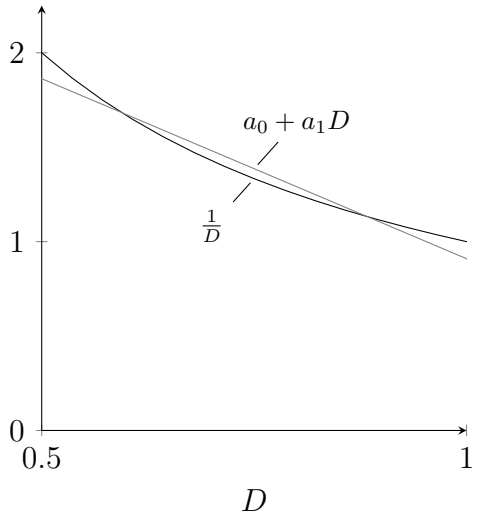


Figure 8.1: Linear least-square approximation of $\frac{1}{D}$ in the region $D \in [0.5, 1]$. The coefficients are $a_0 = 2.8162$ and $a_1 = -1.9066$.

by $\epsilon_{i+1} = \epsilon_i^2$ at subsequent iterations. Thus, finding an initial iterate that is close to the solution $\frac{1}{D}$ *significantly* increases the convergence, since a small initial error (given by $\epsilon_0 = 1 - Dz_0$) will rapidly diminish at subsequent iterations. To generate an initial solution z_0, consider the linear least-squares approximation problem

$$\min_{a_0, a_1} \int_{0.5}^{1} \left(\frac{1}{D} - (a_0 + a_1 D) \right)^2 \mathrm{d}D. \tag{8.7}$$

The (linear) least-squares problem of (8.7) involves calculating the coefficients a_0 and a_1 so that the first-order polynomial $a_0 + a_1 D$ best approximates $\frac{1}{D}$ (in the least-square sense) in the region $[0.5, 1]$. According to [69, Section 8.2], the coefficients a_0 and a_1 are the (unique) solutions to the system of linear equations[1]

$$\begin{aligned} a_0 \int_{0.5}^{1} 1 \mathrm{d}D + a_1 \int_{0.5}^{1} D \mathrm{d}D &= \int_{0.5}^{1} \frac{1}{D} \mathrm{d}D \\ a_0 \int_{0.5}^{1} D \mathrm{d}D + a_1 \int_{0.5}^{1} D^2 \mathrm{d}D &= \int_{0.5}^{1} 1 \mathrm{d}D. \end{aligned}$$

After solving the trivial integrals, the solutions to the system of linear equations are $a_0 = 2.8162$ and $a_1 = -1.9066$. The linear least-square approximation is shown in Figure 8.1. The initial iterate z_0 of Newton's method is set equal to the linear (least-squares) approximation of $\frac{1}{D}$, $z_0 = a_0 + a_1 D$. Note that these (static) coefficients always result in the best linear approximation (in the least-square sense) for the initial iterate z_0, since D is *always* in the interval $[0.5, 1]$. With the initial iterate $z_0 = a_0 + a_1 D$, only two to three iterations of (8.6) are (typically) required for sufficient accuracy.

[1] For an nth order polynomial, the coefficients a_k are determined by solving for $\sum_{k=0}^{n} a_k \int_{0.5}^{1} D^{j+k} \mathrm{d}D = \int_{0.5}^{1} D^j \frac{1}{D} \mathrm{d}D$ for each $j = 0, 1, \ldots, n$.

8.3 Efficient Projection onto the Feasible Set

In this section, the QP

$$\boldsymbol{\Lambda}_{\text{opt}} = \arg\min_{\boldsymbol{\Lambda}} \quad \tfrac{1}{2}\boldsymbol{\Lambda}^{\mathrm{T}}\boldsymbol{H}\boldsymbol{\Lambda} + \boldsymbol{c}^{\mathrm{T}}\boldsymbol{\Lambda} \tag{8.8a}$$

$$\text{subject to } \boldsymbol{A}\boldsymbol{\Lambda} \leq \boldsymbol{b} \tag{8.8b}$$

is solved using the gradient projection method, where (5.54) is repeated here for convenience. The constraint matrices $\boldsymbol{A} \in \mathbb{R}^{(n_{sw}+3)\times n_{sw}}$ and $\boldsymbol{b} \in \mathbb{R}^{n_{sw}+3}$ are defined (from Section 5.5.3) as

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{A}_a & & \\ & \boldsymbol{A}_b & \\ & & \boldsymbol{A}_c \end{bmatrix} \text{ and } \boldsymbol{b} = \begin{bmatrix} \boldsymbol{b}_a \\ \boldsymbol{b}_b \\ \boldsymbol{b}_c \end{bmatrix},$$

respectively, with the per-phase constraint matrix and vector

$$\boldsymbol{A}_p = \begin{bmatrix} \frac{1}{\Delta u_{p,1}} & 0 & 0 & \cdots & 0 & 0 \\ -\frac{1}{\Delta u_{p,1}} & \frac{1}{\Delta u_{p,2}} & 0 & \cdots & 0 & 0 \\ 0 & -\frac{1}{\Delta u_{p,2}} & \frac{1}{\Delta u_{p,3}} & & 0 & 0 \\ \vdots & & & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & -\frac{1}{\Delta u_{p,n_p-1}} & \frac{1}{\Delta u_{p,n_p}} \\ 0 & 0 & 0 & \cdots & 0 & \frac{1}{\Delta u_{p,n_p}} \end{bmatrix}$$

and

$$\boldsymbol{b}_p = \begin{bmatrix} t'_{p,1} & t'_{p,2} - t'_{p,1} & \cdots & T_p - t'_{p,n_p} \end{bmatrix}^{\mathrm{T}},$$

respectively.

In Section 8.3.1, the gradient projection method is employed to solve the QP of (8.8). The decision variable is transformed to the modified switching instants, which enables an efficient projection onto the feasible set. Section 8.3.2 then shows how the projection onto a truncated monotone cone can be efficiently realized.

8.3.1 The Gradient Projection Method

To solve (8.8) on an FPGA, the gradient projection method from Section 2.3.2 is employed. The kth iteration of the method, with a stepsize $s = \frac{1}{\bar{L}_c}$ and the gradient $\nabla f(\boldsymbol{\Lambda}_k) = \boldsymbol{H}\boldsymbol{\Lambda}_k + \boldsymbol{c}$, is

$$\boldsymbol{\Lambda}_{k+1} = \pi_{\mathcal{Z}}\left(\boldsymbol{\Lambda}_k - \frac{1}{\bar{L}_c}\left(\boldsymbol{H}\boldsymbol{\Lambda}_k + \boldsymbol{c}\right)\right) \tag{8.9}$$

where the feasible set is $\mathcal{Z} = \{\boldsymbol{z} : \boldsymbol{A}\boldsymbol{z} \leq \boldsymbol{b}\}$. The first step of the gradient projection method, taking the (unconstrained) step

$$\boldsymbol{\Lambda}_k - \frac{1}{\bar{L}_c}\left(\boldsymbol{H}\boldsymbol{\Lambda}_k + \boldsymbol{c}\right) \tag{8.10}$$

is trivial to realize. The second step, projecting the unconstrained step onto the set $\mathcal{Z}$, requires special attention to efficiently implement.

Transforming the Decision Variable for an Efficient Projection

Recall from Section 5.5.3 that the constraints of (8.8b) can be defined in terms of the modified switching times as

$$0 \leq t_{p,1} \leq t_{p,2} \leq \cdots \leq t_{p,n_p} \leq T_p \tag{8.11}$$

for all $p \in \{a, b, c\}$. The impulse strengths (which are the elements in the strength vector $\boldsymbol{\Lambda}$) can be written in terms of the modified switching instants as

$$t_{p,i} = t'_{p,i} + \frac{\lambda_{p,i}}{\Delta u_{p,i}} \tag{8.12}$$

(recall that $\Delta u_{p,i} \in \{-1, 1\}$, no division is required). Once an unconstrained step has been taken according to (8.10), the impulse strengths are transformed to modified switching transitions according to (8.12).

In the next section, it is shown how to efficiently project onto the set of (8.11). Note the constraints of a phase are independent of the constraints in another phase; the projection of each phase can be considered separately. Thus, only the projection of a single phase is considered from here on in. Denote with $\boldsymbol{t}_p \in \mathbb{R}^{n_p}$ the vector of modified switching transitions that is to be projected.

8.3.2 Efficient Projection onto a Truncated Monotone Cone

The constraints of (8.11) form a so-called *truncated monotone cone*, which in its general form is

$$\bar{\mathbb{K}} = \{\boldsymbol{z} : \underline{z} \leq z_1 \leq z_2 \leq \cdots \leq z_{n_z} \leq \bar{z}\}, \tag{8.13}$$

where $\underline{z}$ and $\bar{z}$ are the lower and upper bounds, respectively.[2] As noted in [70], a truncated monotone cone can be written as the intersection between a (convex) monotone cone and a box,

$$\bar{\mathbb{K}} = \mathbb{K} \cap \mathbb{B},$$

where

$$\mathbb{K} = \{\boldsymbol{z} : z_1 \leq z_2 \leq \cdots \leq z_{n_z}\}$$
$$\mathbb{B} = \{\boldsymbol{z} : \underline{z} \leq z_i \leq \bar{z} \text{ for } i = 1, 2, \ldots, n_z\}$$

are the monotone cone and box, respectively. It is shown according to [70, Theorem 1] that the projection of $\boldsymbol{t}_p$ onto a truncated monotone cone is equivalent to first projecting it onto the monotone cone and then onto the box,

$$\pi_{\bar{\mathbb{K}}}(\boldsymbol{t}_p) = \pi_{\mathbb{B}}(\pi_{\mathbb{K}}(\boldsymbol{t}_p)). \tag{8.14}$$

The second projection, onto the box $\mathbb{B}$, is extremely simple to realize: the projection of the ith component of $\boldsymbol{\xi}$ is simply

$$[\pi_{\mathbb{B}}(\boldsymbol{\xi})]_i = \min\{\max\{\xi_i, \underline{z}\}, \bar{z}\}. \tag{8.15}$$

The first projection onto the monotone cone $\mathbb{K}$ is significantly more complex to realize, and is explained next.

[2] In the case of (8.11), the lower and upper bounds are $\underline{z} = 0$ and $\bar{z} = T_p$, respectively.

Projecting onto a Monotone Cone

Note that a monotone cone can be defined in matrix form as $\mathbb{K} = \{\boldsymbol{z} : \boldsymbol{C}\boldsymbol{z} \leq \boldsymbol{0}_{n_z-1}\}$, where $\boldsymbol{C} \in \mathbb{R}^{(n_z-1)\times n_z}$ is a first-order difference matrix,

$$\boldsymbol{C} = \begin{bmatrix} 1 & -1 & 0 & \cdots & 0 \\ 0 & 1 & -1 & & 0 \\ \vdots & & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 & -1 \end{bmatrix}.$$

Although no closed-form expression exists for the projection onto a monotone cone, an exact solution can be iteratively obtained (see [71]). However, the exact solution is computationally expensive, requiring pseudoinverses of matrices. Therefore, [70] recommended that the projection should be approximated if a computationally efficient option is required. As shown in (2.13), the approximated projection follows as

$$\pi_{\mathbb{K}}(\boldsymbol{t}_p) = \boldsymbol{t}_p - \boldsymbol{C}^{\mathrm{T}}\boldsymbol{\eta}_{\text{opt}} \tag{8.16}$$

with

$$\boldsymbol{\eta}_{\text{opt}} = \arg\min_{\boldsymbol{\eta}\geq\boldsymbol{0}} \tfrac{1}{2}\boldsymbol{\eta}^{\mathrm{T}}\boldsymbol{C}\boldsymbol{C}^{\mathrm{T}}\boldsymbol{\eta} - (\boldsymbol{C}\boldsymbol{y})^{\mathrm{T}}\boldsymbol{\eta}, \tag{8.17}$$

where $\boldsymbol{\eta} \in \mathbb{R}^{n_p-1}$ is the Lagrange multiplier. In order to solve (8.17) to a certain accuracy, the gradient projection method can be used; refer to the gradient projection that is employed to solve (8.17) as the *inner* gradient method, whereas (8.9) is referred to as the *outer* gradient method. It is shown with [70, Proposition 2] that the largest L_d and smallest μ_d eigenvalues of the (dual) Hessian $\boldsymbol{C}\boldsymbol{C}^{\mathrm{T}} \in \mathbb{R}^{(n_p-1)\times(n_p-1)}$, which is a second-order difference matrix,

$$\boldsymbol{C}\boldsymbol{C}^{\mathrm{T}} = \begin{bmatrix} 2 & -1 & 0 & 0 & \cdots & 0 \\ -1 & 2 & -1 & 0 & \cdots & 0 \\ 0 & -1 & 2 & -1 & \cdots & 0 \\ \vdots & & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & -1 & 2 & -1 \\ 0 & 0 & \cdots & 0 & -1 & 2 \end{bmatrix},$$

always satisfy

$$L_d + u_d = 4.$$

This results in an optimal (fixed) step size of $s = \frac{2}{L_d+u_d} = \frac{1}{2}$ for the inner gradient method; thus, the kth (inner) iteration is

$$\boldsymbol{\eta}_{k+1} = \max\left\{\boldsymbol{0}_{n_p-1}, \boldsymbol{\eta}_k - \tfrac{1}{2}(\boldsymbol{C}\boldsymbol{C}^{\mathrm{T}}\boldsymbol{\eta}_k - \boldsymbol{C}\boldsymbol{t}_p)\right\}. \tag{8.18}$$

Importantly, an iteration of (8.18) only requires additions, subtractions, and bit shifts; no multiplications are required. Furthermore, the conditioning number of the Hessian $\boldsymbol{C}\boldsymbol{C}^{\mathrm{T}}$ of (8.17) is very low for the dimensions required by the small-signal controller: for $n_p = 5$ (five switching transitions for a given phase), the conditioning number is below 10.[3] Due to the low conditioning number, only a few iterations of the (inner) gradient method are required to achieve a sufficient accuracy. It is noted in [70] that if warmstarting is employed from the last iterate $\boldsymbol{\eta}_{k+1}$ for each outer iteration of (8.9), only a single (inner) iteration of (8.18) is sufficient.

[3] The conditioning number for the dual Hessian is $\frac{L_d}{\mu_d}$, where $L_d = 2 - 2\cos\left(\frac{(n_p-1)\pi}{n_p}\right)$ and $\mu_d = 2 - 2\cos\left(\frac{\pi}{n_p}\right)$ [70, Proposition 2].

To summarize, an iteration of the gradient projection method of (8.9) involves the following steps. Once an unconstrained step has been taken according to (8.10), the impulse strengths are transformed to modified switching transitions according to (8.12). Then, the vector of modified switching transitions $\boldsymbol{t}_p$ is projected onto the monotone cone with (8.16), which is then projected onto the box with (8.15). The modified switching transitions are then transformed back to impulse strengths according to

$$\lambda_{p,i} = (t_{p,i} - t'_{p,i})\Delta u_{p,i}$$

for the next (outer) iteration of the gradient projection method.

8.4 Implementation and Verification

This section discusses the implementation of the control algorithm on an FPGA. Specifically, Section 8.4.1 gives a summary of the implementation of the control algorithm; this includes the design choices, resource usage, and the time required to execute the control algorithm. Section 8.4.2 then verifies the control algorithm in real-time via a hardware-in-the-loop (HIL) simulation.

8.4.1 Design Choices and Implementation

The entire control algorithm is implemented in very high speed integrated circuit hardware description language (VHDL). Although details regarding VHDL are omitted for the sake of brevity, the interested reader is referred to the excellent textbook [72] on how to use VHDL effectively.

Design Choices

The entire design of the control algorithm uses a single clock, which is a recommended design practice (see [72, Section 19.3.2]). It is decided that the design is clocked using a 50 MHz clock. For a controller with a sampling interval of $T_s = 25\,\mu$s, this results in 1250 clock cycles being available to execute the control algorithm. A low-cost Xilinx XC7Z020 Zynq-7000 FPGA is used, which has 220 DSPE41 slices; each DSP slice contains a single 25×18 bit multiplier.[4] Next, some of the design choices of the control algorithm are discussed.

First, fixed-point arithmetic is used. When compared to floating-point arithmetic, fixed-point arithmetic requires less resources and results in faster computations. Furthermore, there is little reason to use floating-point arithmetic when considering the fact that model inaccuracies and quantization errors (from analogue-to-digital converters) are present in a practical system. Floating-point arithmetic may only be required if certain aspects of a control algorithm require high numerical precision for stability purposes.

Second, all variables are limited to a word length of 18 bits to minimize the usage of DSP slices. Using word lengths that exceed the capabilities of a single DSP slice requires additional DSP slices to be invoked. Furthermore, 18 bits yielded sufficient numerical accuracy.

Third, special attention is given on how the DSP slices are utilized in the design. It is possible to multiply and use the answer within a *single* clock cycle. Doing so, however, will severely limit the clock speed the DSP slices can be operated at (and consequently the clock speed of the entire design). In order to operate a DSP slice at a high clock speed, a technique known as *pipelining* should be used; the result of a DSP slice is passed through a so-called *pipeline register* before it is used. For an illustrative example, refer to Figure 8.2, where the multiplication $y = x_1x_2x_3x_4$ is

[4] Operations such as adding, subtracting, or bit shifting (which relates to multiplication or division by powers of two) typically require little resources to realize and can be implemented using (cheap) logic elements. On the other hand, multiplication is realized by the dedicated multipliers (DSP slices) within the FPGA.

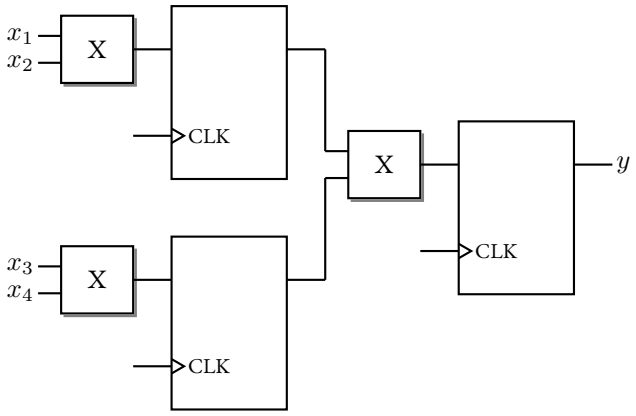


Figure 8.2: Example of pipelining.

shown. As seen, pipeline registers are placed at the output of each DSP slice.[5] Note that two clock cycles are required to *ready* the pipeline registers before the result y is available.

Fourth, the matrix exponentials are calculated offline, at a time resolution of 1 µs, and stored in lookup tables. Furthermore, the steady-state trajectories are also calculated offline, over one fundamental period, and stored in lookup tables.

Finally, recall that the problem size (the dimension of the decision variable $\mathbf{\Lambda}$) is time-varying, and given as $n_{\text{sw}} = n_a + n_b + n_c$ (the total number of switching transitions that fall within the prediction horizon T_p). The design on the FPGA must be of a fixed size, since the hardware required for the design has to be invoked during synthesis; once the FPGA is programmed, additional hardware cannot be invoked. The (fixed-size) implementation assumes a maximum of five switching transitions per phase at any given moment, leading a maximum problem dimension of 15. For a pulse number of $d = 5$ and a prediction horizon of $T_p = 2\,\text{ms}$, there are usually no more than three switching transitions per phase.

DSP Slice Usage and Execution Time

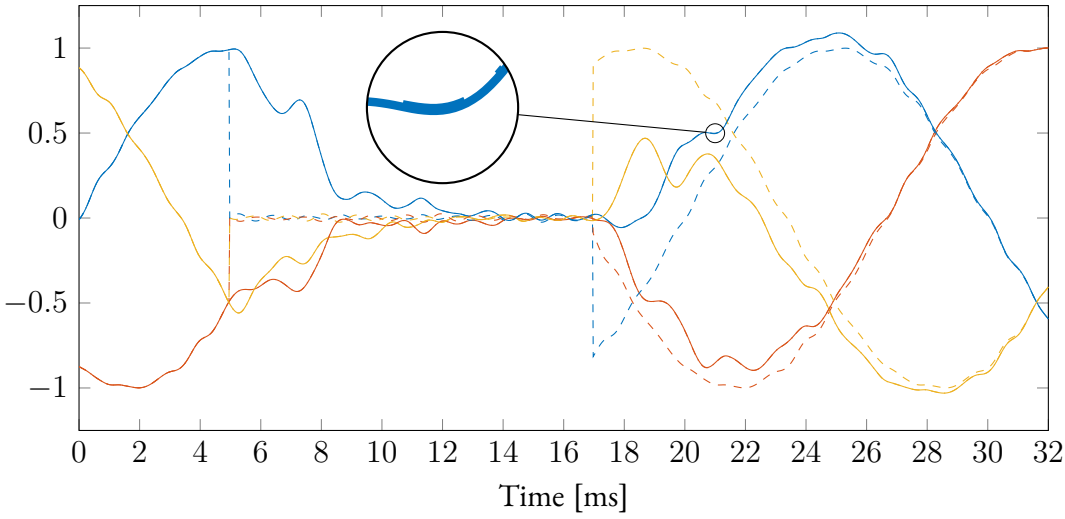
Out of the 220 DSP slices that are available, only 69 were required for the design: 50 for calculating the Hessian and vector, 3 for determining the stepsize, and 16 for the gradient projection method.

To calculate the Hessian and vector, 124 clock cycles are required. Calculating the stepsize (using 3 iterations of Newton's method) requires 24 clock cycles. Solving the QP with the gradient projection method (using 35 iterations) requires 921 clock cycles. In total, 1072 (out of the available 1250) clock cycles are required to execute the entire control algorithm (including determining the incumbent pulse pattern and readying the switching transition within the sampling interval), translating to only 21.4 µs.

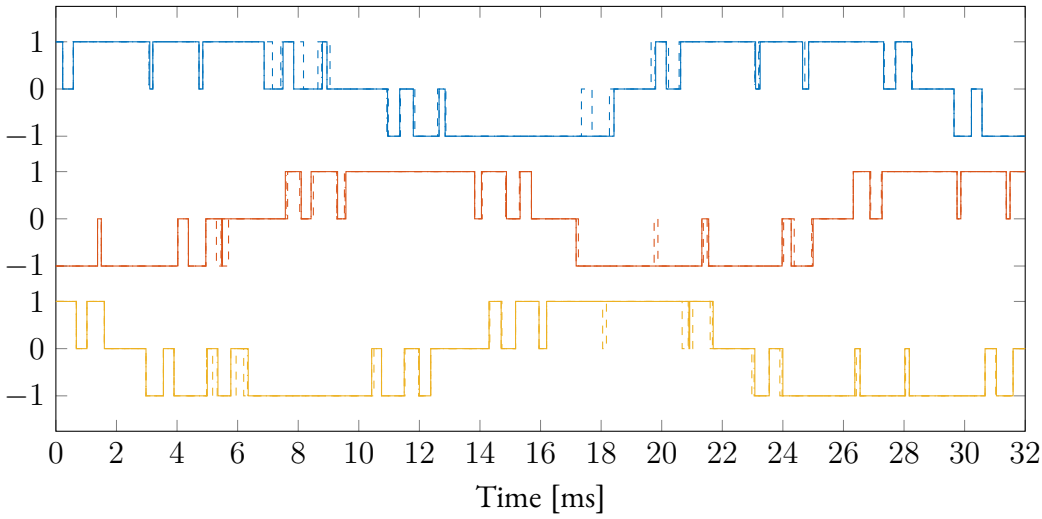
8.4.2 Verification

To validate the implementation, a HIL simulation is considered. The HIL simulation is realized using a discrete-time state-space representation, which is implemented using the remaining resources on the FPGA. Fixed-point variables with a word length of 32 bits are used to accurately simulate the plant. Strictly speaking, the HIL simulation should execute in parallel with the control algorithm to represent a physical plant. However, the plant in Matlab is not simulated while the control algorithm is executing; only once the control algorithm has executed is the plant simulated for a time of T_s.

[5] Additional pipeline registers may also be used, which may result in even higher clock speeds. However, a single pipeline register at the output enabled a DSP slice to be operated at 50 MHz.



(a) Grid currents



(b) Switch positions

Figure 8.3: The grid current (in pu) responses during reference steps of the FPGA- and Matlab-implemented controllers. References are indicated by dashed lines. The responses of the FPGA- and Matlab-implemented controllers are indicated with solid and dash-dotted lines, respectively.

Thus, for comparison purposes to the Matlab simulation, the HIL simulation also executes *after* the control algorithm has executed.

The sampling interval and prediction horizon of the controller are set to $T_s = 25\,\mu s$ and $T_p = 2\,ms$, respectively. All state variables are penalized equally with $\boldsymbol{Q} = \mathbf{I}_6$ and all switching transition modifications are also penalized equally with $\boldsymbol{R} = 2\mathbf{I}_{n_{sw}}$. A pulse number of $d = 5$ is used. The parameters of the grid-connected converter system can be found in Section 3.3.2.

In Figure 8.3, the response of the grid current during reference steps is shown. Initially, the converter is operating at rated power with unity power factor. The (real) power reference is stepped to $P^* = 0\,pu$ at 5 ms, and then back to $P^* = 1\,pu$ at 17 ms. As seen, the simulation on the FPGA is near identical to that of Matlab, demonstrating that the controller can execute in real-time within a short sampling interval. The largest errors in switching times (between that of Matlab and the FPGA) for phases a, b, and c are approximately 3.3 µs, 3.7 µs, and 3.3 µs, respectively.[6] The

[6]The switching times are captured at a resolution of 500 ns from the FPGA.

average errors (in switching times) for phases a, b, and c are approximately 1.3 µs, 1.1 µs, and 1.2 µs, respectively.

8.5 Summary

In this chapter, recommendations were given to reduce the computational burden of the QP underlying the standard small-signal controller, with the purpose of enabling an efficient implementation on an FPGA.

It was shown how the calculations regarding the Hessian and vector can be reduced by exploiting the algebraic structure of the control problem and the characteristics of the load. It was further shown that the Lipschitz constant can be overestimated using the infinity-norm of the Hessian, and how the stepsize of the gradient method can be computed using Newton's method. Finally, it was explained how the projection operator of the gradient projection method can be efficiently implemented without requiring any multiplications.

Using the recommendations that were given, an efficient implementation of the standard controller was achieved, as only 69 DSP slices were required. The control algorithm required 21.4 µs to execute. The FPGA-implemented controller was verified via a HIL simulation.

Part III

Summary and Outlook

Chapter 9

Summary and Outlook

This chapter reviews the main results of this thesis. Recommendations for future research are presented.

Chapter Contents

9.1 Main Summaries

Four main results were achieved in this thesis, which are now briefly summarized.

The Small-Signal Controller: A Generalized Model Predictive Pulse Pattern Controller

With the proposed small-signal controller, fast closed-loop control of a higher-order converter system that is modulated with OPPs was achieved. During the steady state, the controller modulated the converter system with the nominal pulse pattern, thus achieving superior harmonic distortions. The small-signal controller is a natural generalization of MP^{3}C, which is a state-of-the-art industrial control technique, to higher-order systems.

Key to the controller was to approximate the modifications of a pulse pattern by the strength of impulses. This resulted in a linear internal dynamic model for the controller, which enabled the underlying optimization problem to be a QP. Furthermore, an optional step (the successive re-linearization around the newly-calculated switching instants) of the control algorithm allowed for a highly accurate prediction of the system states.

Constrained Small-Signal Controller

The formulation of the small-signal controller was extended so that bounds can be imposed on the converter states. These bounds were realized as (linear) constraints, which were added to the underlying QP of the controller. By using the notion of relaxation (or slack) variables gave rise to so-called soft constraints that allowed the optimization problem to always maintain feasibility.

Balancing of the Neutral-Point Potential

Balancing of the neutral-point potential was integrated in the (standard) small-signal controller. An important aspect of the balancing method was to represent the absolute value of a pulse pattern by a so-called absolute pulse pattern. In accordance with the small-signal controller, the strengths of impulses were used to approximate modifications of the absolute pulse pattern. The method was demonstrated to work effectively at the (difficult) operating condition of zero power factor at the converter terminals (where traditional balancing methods fail).

FPGA Implementation of the Standard Controller

It was shown that the standard small-signal controller is a practically viable control method that can be used today, as it was implemented on a low-cost FPGA. Thus fur, none of the other existing OPP-based controllers for higher-order systems have been proven to operate in real-time.

9.2 Proposed Extensions and Additions

Recommendations to extend the research of this thesis are now proposed.

Compensation for the Standard Small-Signal Controller

It was mentioned in Section 5.5.7 that the internal dynamic model of the standard small-signal controller assumed that the nominal and incumbent pulse patterns were equal, $\boldsymbol{u}^*_{abc} = \boldsymbol{u}_{abc}$. This is not always true; switching transitions are often advanced or delayed, which creates a mismatch between these two pulse patterns (see Figure 9.1). This mismatch manifests itself as an error in the prediction of the state vector.

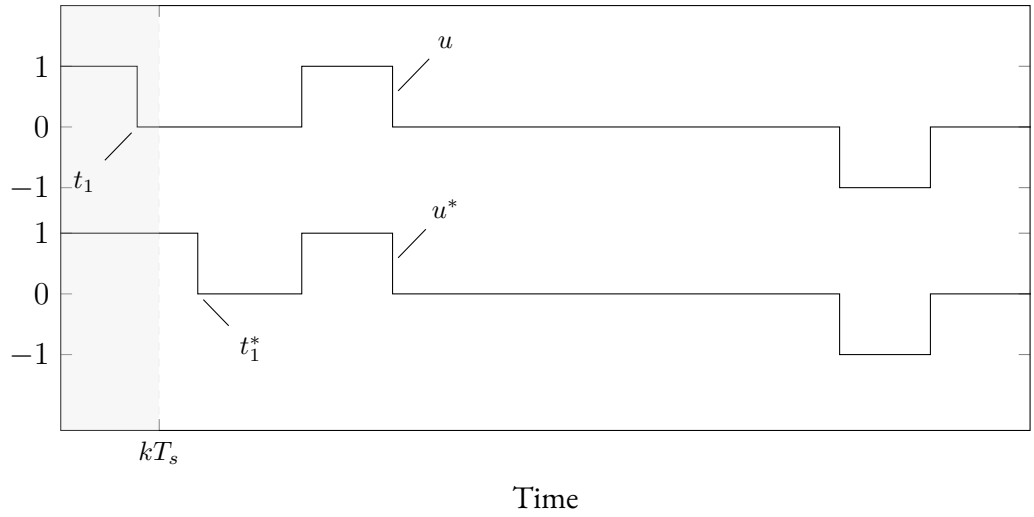
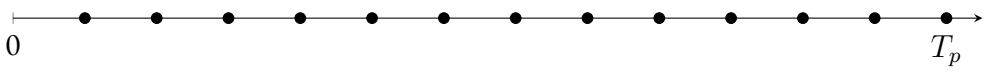


Figure 9.1: The case when the incumbent u and nominal u^* pulse patterns are not equal. Here, the nominal switching transition t_1^* has been advanced to occur at t_1. The current sampling instant is denoted with kT_s.



State-Constraint Prediction Instants

(a) Fixed constraint prediction interval.

State-Constraint Prediction Instants

(b) Variable constraint prediction interval.

Figure 9.2: Illustration of state-constraint prediction instants.

An additional term is thus required in the internal dynamic model of the standard small-signal controller to compensate for this mismatch. As a starting point, it is worthwhile to look at how the advanced small-signal controller incorporates the mismatch between the two pulse patterns [see (5.18)]; a similar route can be used for the standard small-signal controller. Note that the mismatch will require the integration of rectangular areas; to reduce the computational burden, the rectangular areas can be approximated by the strengths of impulses.

Move Blocking for State-Constraint Predictions

It was shown in Section 6.3 that shorter constraint prediction intervals T_c (the time between state-prediction instants), generally, increases the performance of the constrained small-signal controller; however, the dimension of the optimization problem also increases.

Typically, the state-constraint prediction instants occur at a fixed interval, see Figure 9.2a. To reduce the dimensions of the problem without significantly deteriorating the performance, the state-constraint prediction instants can be predicted at variable time intervals (instead of a fixed short or long interval). Specifically, as shown in Figure 9.2b, the first few state-constraint prediction instants are finely spaced (say, the first three instants are predicted at a 50 µs interval) whereas later state-constraint prediction instants are coarsely spaced (say, at a 250 µs interval). This is similar to so-called *move blocking* that is used in MPC [73].

Advanced Small-Signal Controller with Integrated Balancing of the Neutral-Point Potential

In Chapter 7, only the standard small-signal controller was adapted to include balancing of the neutral-point potential. Integrating balancing of the neutral-point potential in the advanced small-signal controller will involve similar steps to those used when the standard small-signal controller was considered (no further theory will be required to be developed). However, it is expected that the computations required will increase moderately due to the additional terms in the model of the advanced small-signal controller.

Pulse Insertion for the Small-Signal Controller

As mentioned in Section 5.6.1, it is highly likely that the response time can be reduced if additional switching transitions are inserted during transients. As a starting point to implement pulse insertion, refer to [4, Section 12.6] and [59].

Further Implementations on an FPGA

Only the standard small-signal controller was implemented on an FPGA in Chapter 8. The advanced small-signal controller, constrained small-signal controller, and the small-signal controller with integrated balancing of the neutral-point potential have not been implemented on an FPGA. Given that only 69 (out of the 220) DSP slices were used for the standard small-signal controller, it is reasonable to expect that the other controllers are also implementable. It would be interesting to see how the projection operator of the gradient projection method can be implemented for the constrained small-signal controller, since the feasible set has a nontrivial shape.

Computing the Matrix Exponential Online

The matrix exponential was calculated offline in Chapter 8. However, if for any reason the matrix exponential is required to be calculated online, refer to the methods in [74]. Preliminary simulations show that a Taylor series expansion up to the 5th order in combination with the so-called *scaling-and-squaring method* [74, Method 3] seems to result in sufficient numerical accuracy for the control problem.

9.3 Outlook

Although the small-signal controller has demonstrated promising results in simulations, where the grid is modelled to be ideal, there are certain aspects of a practical converter system that were not considered in this thesis and may need addressing. It is believed that the following topics are important research questions.

Inclusion of Parameter Estimation

It was assumed that the parameters of the grid were known exactly. However, the parameters of a practical grid are not static. Thus, some form of estimation may be required to be integrated within the control algorithm.

Response to Grid Asymmetries

The parameters and the grid voltage were assumed to be symmetrical. It would be interesting to see if the internal dynamic model of the controller can be adapted to compensate for the grid asymmetries, or if it will be required to develop appropriate (nominal) pulse patterns.

Response to Grid Voltage Harmonics

Unknown voltage harmonics may be present at the point of common coupling, which may cause significant harmonics in the converter current. In such a case, the immunity of the controller against these grid (voltage harmonic) disturbances has to be improved. A possibility to address this problem is to identify the voltage harmonics (via online estimation) and include them in the internal dynamic model of the controller.

Appendices

Appendix A

The Dual of the Projection

For details regarding the theory of duality, interested readers are referred to [6, Section 4.4 and Chapter 6] and, for an accessible source, [11, Chapter 12].

The projection operator is defined as

$$\pi_{\mathcal{Z}}(\boldsymbol{x}) = \arg\min_{\boldsymbol{z}\in\mathcal{Z}} \tfrac{1}{2}\|\boldsymbol{z}-\boldsymbol{x}\|_2^2, \tag{A.1}$$

which projects $\boldsymbol{x}$ onto the set $\mathcal{Z}$. Consider the case where $\mathcal{Z}$ is a polyhedron defined as $\mathcal{Z} = \{\boldsymbol{z} : \boldsymbol{A}\boldsymbol{z} \leq \boldsymbol{b}\}$, where $\boldsymbol{A} \in \mathbb{R}^{n_g \times n_z}$ and $\boldsymbol{b} \in \mathbb{R}^{n_g}$ are the constraint matrix and vector, respectively.

The Lagrangian (or Lagrange dual function) is

$$\mathcal{L}(\boldsymbol{z}, \boldsymbol{\eta}) = \tfrac{1}{2}\|\boldsymbol{z}-\boldsymbol{x}\|_2^2 + (\boldsymbol{A}\boldsymbol{z}-\boldsymbol{b})^{\mathrm{T}}\boldsymbol{\eta}, \tag{A.2}$$

where $\boldsymbol{\eta} \in \mathbb{R}^{n_g}$ is the nonnegative Lagrangian multiplier. The dual objective function is defined as

$$d(\boldsymbol{\eta}) = \min_{\boldsymbol{z}} \mathcal{L}(\boldsymbol{z}, \boldsymbol{\eta}).$$

It follows from the stationary condition (where the first-order partial derivatives are zero) that the Lagrangian of (A.2) achieves a minimum at

$$\boldsymbol{z} = \boldsymbol{x} - \boldsymbol{A}^{\mathrm{T}}\boldsymbol{\eta}. \tag{A.3}$$

By substituting (A.3) back into (A.2), the dual objective function becomes

$$\begin{aligned} d(\boldsymbol{\eta}) &= \tfrac{1}{2}\boldsymbol{\eta}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{\eta} + (\boldsymbol{A}(\boldsymbol{x}-\boldsymbol{A}^{\mathrm{T}}\boldsymbol{\eta})-\boldsymbol{b})^{\mathrm{T}}\boldsymbol{\eta} \\ &= -\tfrac{1}{2}\boldsymbol{\eta}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{\eta} + (\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b})^{\mathrm{T}}\boldsymbol{\eta} \end{aligned}$$

The dual problem is to maximize $d(\boldsymbol{\eta})$ subject to $\boldsymbol{\eta} \geq \boldsymbol{0}$, which is equivalent to minimizing $-d(\boldsymbol{\eta})$:

$$\boldsymbol{\eta}_{\text{opt}} = \arg\min_{\boldsymbol{\eta}\geq\boldsymbol{0}} \tfrac{1}{2}\boldsymbol{\eta}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{\eta} + (\boldsymbol{b}-\boldsymbol{A}\boldsymbol{x})^{\mathrm{T}}\boldsymbol{\eta}. \tag{A.4}$$

The primal problem of (A.1) attains its minimum at

$$\boldsymbol{z}_{\text{opt}} = \boldsymbol{x} - \boldsymbol{A}^{\mathrm{T}}\boldsymbol{\eta}_{\text{opt}}. \tag{A.5}$$

Appendix B

Differential Equations of a Grid-Connected Converter

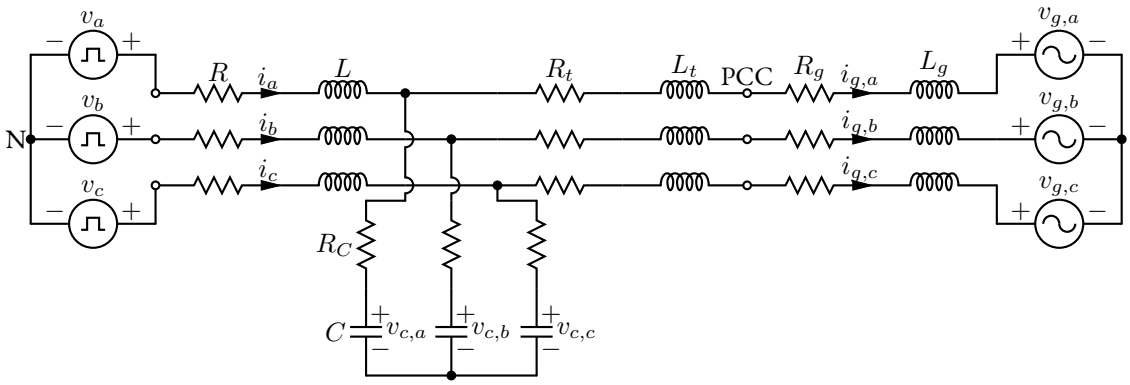


Figure B.1: Grid-connected NPC converter. The transformer is represented by its leakage inductance and resistance. The converter is represented by switched voltage sources.

Recall the reduced Clarke transformation from Section 3.1.2 as

$$\boldsymbol{\xi}_{\alpha\beta} = \boldsymbol{K}\boldsymbol{\xi}_{abc}$$

with

$$\boldsymbol{K} = \frac{2}{3}\begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \end{bmatrix}.$$

From this, ξ_α and ξ_β follow as

$$\begin{aligned} \xi_\alpha &= \tfrac{2}{3}(\xi_a - \tfrac{1}{2}\xi_b - \tfrac{1}{2}\xi_c) \\ &= \tfrac{2}{3}((\xi_a - \xi_b) + \tfrac{1}{2}(\xi_b - \xi_c)) \end{aligned} \tag{B.1}$$

and

$$\xi_\beta = \tfrac{2}{3}\tfrac{\sqrt{3}}{2}(\xi_b - \xi_c), \tag{B.2}$$

respectively.

Denote with $L_{gt} = L_g + L_t$ and $R_{gt} = R_g + R_t$ the inductances and resistances, respectively, of the grid and transformer that are lumped together (see Figure B.1).

Next, the differential equations describing the converter currents, grid currents, capacitor voltages are derived.

Converter Current The following equations can be derived from Figure B.1:

$$v_a - v_b = (R + R_C)(i_a - i_b) + L(\frac{\mathrm{d}i_a}{\mathrm{d}t} - \frac{\mathrm{d}i_b}{\mathrm{d}t}) - R_C(i_{g,a} - i_{g,b}) + (v_{c,a} - v_{c,b}) \tag{B.3a}$$

$$v_b - v_c = (R + R_C)(i_b - i_c) + L(\frac{\mathrm{d}i_b}{\mathrm{d}t} - \frac{\mathrm{d}i_c}{\mathrm{d}t}) - R_C(i_{g,b} - i_{g,c}) + (v_{c,b} - v_{c,c}). \tag{B.3b}$$

Inserting (B.3) into (B.1) results in

$$\begin{aligned} v_\alpha &= \tfrac{2}{3}((R + R_C)(i_a - \tfrac{1}{2}i_b - \tfrac{1}{2}i_c) + L(\frac{\mathrm{d}i_a}{\mathrm{d}t} - \tfrac{1}{2}\frac{\mathrm{d}i_b}{\mathrm{d}t} - \tfrac{1}{2}\frac{\mathrm{d}i_c}{\mathrm{d}t}) - R_C(i_{g,a} - \tfrac{1}{2}i_{g,b} - \tfrac{1}{2}i_{g,c})) \\ &\quad + (v_{c,a} - \tfrac{1}{2}v_{c,b} - \tfrac{1}{2}v_{c,c}) \\ &= (R + R_C)i_\alpha + L\frac{\mathrm{d}i_\alpha}{\mathrm{d}t} - R_C i_{g,\alpha} + v_{c,\alpha}, \end{aligned}$$

from which the evolution of i_α follows as

$$\frac{\mathrm{d}i_\alpha}{\mathrm{d}t} = -\frac{R + R_C}{L}i_\alpha + \frac{R_C}{L}i_{g,\alpha} - \frac{1}{L}v_{c,\alpha} + \frac{1}{L}v_\alpha. \tag{B.4}$$

After inserting (B.3) into (B.2), and by using similar manipulations to those used in the derivation of (B.4), it can be shown that the evolution of i_β follows as

$$\frac{\mathrm{d}i_\beta}{\mathrm{d}t} = -\frac{R + R_C}{L}i_\beta + \frac{R_C}{L}i_{g,\beta} - \frac{1}{L}v_{c,\beta} + \frac{1}{L}v_\beta \tag{B.5}$$

With the definition of $\boldsymbol{i} = [i_a\ i_b]^{\mathrm{T}}$, it follows that

$$\frac{\mathrm{d}\boldsymbol{i}(t)}{\mathrm{d}t} = -\frac{R + R_C}{L}\boldsymbol{i}(t) + \frac{R_C}{L}\boldsymbol{i}_g(t) - \frac{1}{L}\boldsymbol{v}_c(t) + \frac{V_d}{2L}\boldsymbol{K}\boldsymbol{u}_{abc}(t), \tag{B.6}$$

where $\boldsymbol{v}_{\alpha\beta} = \frac{V_d}{2}\boldsymbol{K}\boldsymbol{u}_{abc}$ has been used.

Grid Current The following equations can be derived from Figure B.1:

$$v_a - v_b = R(i_a - i_b) + L(\frac{\mathrm{d}i_a}{\mathrm{d}t} - \frac{\mathrm{d}i_b}{\mathrm{d}t}) + R_{gt}(i_{g,a} - i_{g,b}) + L_{gt}(\frac{\mathrm{d}i_{g,a}}{\mathrm{d}t} - \frac{\mathrm{d}i_b}{\mathrm{d}t}) + (v_{g,a} - v_{g,b}) \tag{B.7a}$$

$$v_b - v_c = R(i_b - i_c) + L(\frac{\mathrm{d}i_b}{\mathrm{d}t} - \frac{\mathrm{d}i_c}{\mathrm{d}t}) + R_{gt}(i_{g,b} - i_{g,c}) + L_{gt}(\frac{\mathrm{d}i_{g,b}}{\mathrm{d}t} - \frac{\mathrm{d}i_c}{\mathrm{d}t}) + (v_{g,b} - v_{g,c}). \tag{B.7b}$$

Inserting (B.7) into (B.1) yields

$$\begin{aligned} v_\alpha &= \tfrac{2}{3}(R(i_a - \tfrac{1}{2}i_b - \tfrac{1}{2}i_c) + L(\frac{\mathrm{d}i_a}{\mathrm{d}t} - \tfrac{1}{2}\frac{\mathrm{d}i_b}{\mathrm{d}t} - \tfrac{1}{2}\frac{\mathrm{d}i_c}{\mathrm{d}t}) \\ &\quad + R_{gt}(i_{g,a} - \tfrac{1}{2}i_{g,b} - \tfrac{1}{2}i_{g,c}) + L_{gt}(\frac{\mathrm{d}i_{g,a}}{\mathrm{d}t} - \tfrac{1}{2}\frac{\mathrm{d}i_{g,b}}{\mathrm{d}t} - \tfrac{1}{2}\frac{\mathrm{d}i_{g,c}}{\mathrm{d}t})) + (v_{g,a} - \tfrac{1}{2}v_{g,b} - \tfrac{1}{2}v_{g,c}) \\ &= Ri_\alpha + L\frac{\mathrm{d}i_\alpha}{\mathrm{d}t} + R_{gt}i_{g,\alpha} + L_{gt}\frac{\mathrm{d}i_{g,\alpha}}{\mathrm{d}t} + v_{g,\alpha}, \end{aligned}$$

and after inserting (B.5),

$$v_\alpha = -R_C i_\alpha + (R_{gt} + R_C)i_{g,\alpha} + L_{gt}\frac{\mathrm{d}i_{g,\alpha}}{\mathrm{d}t} - v_{c,\alpha} + v_\alpha + v_{g,\alpha}.$$

The evolution of $i_{g,\alpha}$ follows as

$$\frac{\mathrm{d}i_{g,\alpha}}{\mathrm{d}t} = \frac{R_C}{L_{gt}} i_\alpha - \frac{R_{gt}+R_C}{L_{gt}} i_{g,\alpha} + \frac{1}{L_{gt}} v_{c,\alpha} - \frac{1}{L_{gt}} v_{g,\alpha}. \tag{B.8}$$

After inserting (B.7) into (B.2), and by using similar manipulations to those used in the derivation of (B.8), it can be shown that the evolution of $i_{g,\beta}$ follows as

$$\frac{\mathrm{d}i_{g,\beta}}{\mathrm{d}t} = \frac{R_C}{L_{gt}} i_\beta - \frac{R_{gt}+R_C}{L_{gt}} i_{g,\beta} + \frac{1}{L_{gt}} v_{c,\beta} - \frac{1}{L_{gt}} v_{g,\beta}. \tag{B.9}$$

With the definition $\boldsymbol{i}_g = [i_{g,\alpha}\ i_{g,\beta}]^{\mathrm{T}}$, it follows that

$$\frac{\mathrm{d}\boldsymbol{i}_g(t)}{\mathrm{d}t} = \frac{R_C}{L_{gt}} \boldsymbol{i}(t) - \frac{R_{gt}+R_C}{L_{gt}} \boldsymbol{i}_g(t) + \frac{1}{L_{gt}} \boldsymbol{v}_c(t) - \frac{1}{L_{gt}} \boldsymbol{v}_g(t). \tag{B.10}$$

Capacitor Voltage The following equations can be derived from Figure B.1

$$\begin{aligned} C\frac{\mathrm{d}v_{c,a}}{\mathrm{d}t} &= i_a - i_{g,a} \\ C\frac{\mathrm{d}v_{c,b}}{\mathrm{d}t} &= i_b - i_{g,b} \\ C\frac{\mathrm{d}v_{c,c}}{\mathrm{d}t} &= i_c - i_{g,c}, \end{aligned}$$

which lead to

$$\frac{\mathrm{d}v_{c,a}}{\mathrm{d}t} - \frac{\mathrm{d}v_{c,b}}{\mathrm{d}t} = \frac{1}{C}(i_a - i_b) - \frac{1}{C}(i_{g,a} - i_{g,b}) \tag{B.11a}$$

$$\frac{\mathrm{d}v_{c,b}}{\mathrm{d}t} - \frac{\mathrm{d}v_{c,c}}{\mathrm{d}t} = \frac{1}{C}(i_b - i_c) - \frac{1}{C}(i_{g,b} - i_{g,c}). \tag{B.11b}$$

Inserting (B.11) into (B.1) leads to the evolution of $v_{c,\alpha}$ as

$$\begin{aligned} \frac{\mathrm{d}v_{c,\alpha}}{\mathrm{d}t} &= \tfrac{2}{3}\left(\frac{1}{C}(i_a - \tfrac{1}{2}i_b - \tfrac{1}{2}i_c) - \frac{1}{C}(i_{g,a} - \tfrac{1}{2}i_{g,b} - \tfrac{1}{2}i_{g,c})\right) \\ &= \frac{1}{C} i_\alpha - \frac{1}{C} i_{g,\alpha}. \end{aligned} \tag{B.12}$$

Similarly, after inserting (B.11) into (B.2), the evolution of $v_{c,\beta}$ is

$$\begin{aligned} \frac{\mathrm{d}v_{c,\beta}}{\mathrm{d}t} &= \tfrac{2}{3}\tfrac{\sqrt{3}}{2}\left(\frac{1}{C}(i_b - i_c) - \frac{1}{C}(i_{g,b} - i_{g,c}) + (v_{c,b} - v_{c,c})\right) \\ &= \frac{1}{C} i_\beta - \frac{1}{C} i_{g,\beta} \end{aligned} \tag{B.13}$$

With the definition $\boldsymbol{v}_c = [v_{c,\alpha}\ v_{c,\beta}]^{\mathrm{T}}$, it follows that

$$\frac{\mathrm{d}\boldsymbol{v}_c(t)}{\mathrm{d}t} = \frac{1}{C}\boldsymbol{i}(t) - \frac{1}{C}\boldsymbol{i}_g(t). \tag{B.14}$$

Appendix C

The Step and Impulse Functions

The (unit) *step function*, also known as the Heavyside function, is defined as

$$h(t-t_i) = \begin{cases} 1 & \text{if } t \geq t_i \\ 0 & \text{else.} \end{cases} \tag{C.1}$$

A useful feature of the step function is that it changes the lower bounds of an integral,

$$\int_{-\infty}^{t} h(t-t_i) f(t)\, \mathrm{d}t = \int_{t_i}^{t} f(t)\, \mathrm{d}t\, h(t-t_i). \tag{C.2}$$

The derivative of the step function,

$$\frac{\mathrm{d}h(t-t_i)}{\mathrm{d}t} = \delta(t-t_i),$$

is known as the *impulse function* (or the Dirac Delta function). The impulse function $\lambda\delta(t-t_i)$ is defined to have infinite amplitude and zero duration,

$$\lambda\delta(t-t_i) = \begin{cases} \infty & \text{if } t = t_i \\ 0 & \text{else,} \end{cases} \tag{C.3}$$

and its integral is equal to its *strength* λ,

$$\int_{-\infty}^{t} \lambda\delta(t-t_i)\, \mathrm{d}t = \lambda h(t-t_i). \tag{C.4}$$

An extremely useful property of the impulse is the so-called *sifting property*,

$$\int_{-\infty}^{t} \delta(t-t_i) f(t)\, \mathrm{d}t = f(t_i) h(t-t_i), \tag{C.5}$$

where $f(t)$ is assumed to be continuous at t_i.

Note that the impulse is not a function in the traditional sense: it is a *generalized function* (also known as a distribution), and is defined by its integral. For further reading on the impulse function, refer to [26, Section 12.3] and [75, Section 4.5].

Appendix D

Manipulations Involving the Rectangle Input

D.1 Expanding the Rectangle Input

Recall that the rectangle input is defined as

$$\bar{\boldsymbol{u}}_{abc}(t) = \boldsymbol{u}_{abc}(t) - \boldsymbol{u}^*_{abc}(t),$$

where

$$\boldsymbol{u}_{abc}(t) = \begin{bmatrix} u_a(t) \\ u_b(t) \\ u_c(t) \end{bmatrix} = \begin{bmatrix} u_{a,0} + \sum_{i=1}^{n_a} \Delta u_{a,i} h(t - t'_{a,i}) \\ u_{b,0} + \sum_{i=1}^{n_b} \Delta u_{b,i} h(t - t'_{b,i}) \\ u_{c,0} + \sum_{i=1}^{n_c} \Delta u_{c,i} h(t - t'_{c,i}) \end{bmatrix}$$

and

$$\boldsymbol{u}^*_{abc}(t) = \begin{bmatrix} u^*_a(t) \\ u^*_b(t) \\ u^*_c(t) \end{bmatrix} = \begin{bmatrix} u^*_{a,0} + \sum_{i=1}^{n^*_a} \Delta u^*_{a,i} h(t - t^*_{a,i}) \\ u^*_{b,0} + \sum_{i=1}^{n^*_b} \Delta u^*_{b,i} h(t - t^*_{b,i}) \\ u^*_{c,0} + \sum_{i=1}^{n^*_c} \Delta u^*_{c,i} h(t - t^*_{c,i}) \end{bmatrix}.$$

The expression $\boldsymbol{\Gamma}$ defined in (5.30), which appears in (5.31), can be re-arranged as

$$\begin{aligned}
\boldsymbol{\Gamma}(t) = &\sum_{i=1}^{n_a} \left(e^{\boldsymbol{F}(t-t'_{a,i})} - \mathbf{I}_{n_x} \right) \boldsymbol{F}^{-1} \boldsymbol{G}_a \Delta u_{a,i} h(t - t'_{a,i}) \\
&+ \sum_{i=1}^{n_b} \left(e^{\boldsymbol{F}(t-t'_{b,i})} - \mathbf{I}_{n_x} \right) \boldsymbol{F}^{-1} \boldsymbol{G}_b \Delta u_{b,i} h(t - t'_{b,i}) \\
&+ \sum_{i=1}^{n_c} \left(e^{\boldsymbol{F}(t-t'_{c,i})} - \mathbf{I}_{n_x} \right) \boldsymbol{F}^{-1} \boldsymbol{G}_c \Delta u_{c,i} h(t - t'_{c,i}) \\
&- \sum_{i=1}^{n^*_a} \left(e^{\boldsymbol{F}(t-t^*_{a,i})} - \mathbf{I}_{n_x} \right) \boldsymbol{F}^{-1} \boldsymbol{G}_a \Delta u^*_{a,i} h(t - t^*_{a,i}) \\
&- \sum_{i=1}^{n^*_b} \left(e^{\boldsymbol{F}(t-t^*_{b,i})} - \mathbf{I}_{n_x} \right) \boldsymbol{F}^{-1} \boldsymbol{G}_b \Delta u^*_{b,i} h(t - t^*_{b,i}) \\
&- \sum_{i=1}^{n^*_c} \left(e^{\boldsymbol{F}(t-t^*_{c,i})} - \mathbf{I}_{n_x} \right) \boldsymbol{F}^{-1} \boldsymbol{G}_c \Delta u^*_{c,i} h(t - t^*_{c,i}) \\
&+ \left(e^{\boldsymbol{F}t} - \mathbf{I}_{n_x} \right) \boldsymbol{F}^{-1} \boldsymbol{G} \Delta \boldsymbol{u}_{abc,0},
\end{aligned} \tag{D.1}$$

where the fact has been used that an integral involving the step function is

$$\int_0^t \boldsymbol{e}^{\boldsymbol{F}(t-\tau)}\boldsymbol{G}_p\Delta u_{p,i}h(\tau - t'_{p,i})\,\mathrm{d}\tau = \left(\boldsymbol{e}^{\boldsymbol{F}(t-t'_{p,i})} - \mathbf{I}_{n_x}\right)\boldsymbol{F}^{-1}\boldsymbol{G}_p\Delta u_{p,i}h(t - t'_{p,i}).$$

Furthermore, $\Delta\boldsymbol{u}^{\mathrm{T}}_{abc,0}$ has been introduced as $\Delta\boldsymbol{u}^{\mathrm{T}}_{abc,0} = [u_{a,0}\ u_{b,0}\ u_{c,0}] - [u^*_{a,0}\ u^*_{b,0}\ u^*_{c,0}]$.

For convenience, (D.1) is decomposed into three terms,

$$\boldsymbol{\Gamma}(t) = \boldsymbol{\Gamma}'(t) - \boldsymbol{\Gamma}^*(t) + \boldsymbol{\Gamma}_0(t), \tag{D.2}$$

where

$$\begin{aligned}
\boldsymbol{\Gamma}'(t) = &\sum_{i=1}^{n_a}\left(\boldsymbol{e}^{\boldsymbol{F}(t-t'_{a,i})} - \mathbf{I}_{n_x}\right)\boldsymbol{F}^{-1}\boldsymbol{G}_a\Delta u_{a,i}h(t - t'_{a,i}) \\
&+ \sum_{i=1}^{n_b}\left(\boldsymbol{e}^{\boldsymbol{F}(t-t'_{b,i})} - \mathbf{I}_{n_x}\right)\boldsymbol{F}^{-1}\boldsymbol{G}_b\Delta u_{b,i}h(t - t'_{b,i}) \\
&+ \sum_{i=1}^{n_c}\left(\boldsymbol{e}^{\boldsymbol{F}(t-t'_{c,i})} - \mathbf{I}_{n_x}\right)\boldsymbol{F}^{-1}\boldsymbol{G}_c\Delta u_{c,i}h(t - t'_{c,i})
\end{aligned} \tag{D.3}$$

$$\begin{aligned}
\boldsymbol{\Gamma}^*(t) = &\sum_{i=1}^{n^*_a}\left(\boldsymbol{e}^{\boldsymbol{F}(t-t^*_{a,i})} - \mathbf{I}_{n_x}\right)\boldsymbol{F}^{-1}\boldsymbol{G}_a\Delta u^*_{a,i}h(t - t^*_{a,i}) \\
&+ \sum_{i=1}^{n^*_b}\left(\boldsymbol{e}^{\boldsymbol{F}(t-t^*_{b,i})} - \mathbf{I}_{n_x}\right)\boldsymbol{F}^{-1}\boldsymbol{G}_b\Delta u^*_{b,i}h(t - t^*_{b,i}) \\
&+ \sum_{i=1}^{n^*_c}\left(\boldsymbol{e}^{\boldsymbol{F}(t-t^*_{c,i})} - \mathbf{I}_{n_x}\right)\boldsymbol{F}^{-1}\boldsymbol{G}_c\Delta u^*_{c,i}h(t - t^*_{c,i})
\end{aligned} \tag{D.4}$$

$$\boldsymbol{\Gamma}_0(t) = \left(\boldsymbol{e}^{\boldsymbol{F}t} - \mathbf{I}_{n_x}\right)\boldsymbol{F}^{-1}\boldsymbol{G}\Delta\boldsymbol{u}_{abc,0}. \tag{D.5}$$

D.2 Quadratic Objective Function Terms Involving the Rectangle Input

For convenience and clarity, the integrals of the terms in (5.46) are considered separately,

$$\begin{aligned}
\boldsymbol{c}^{\mathrm{T}}_{\Gamma,i'} &= \int_0^{T_p}\boldsymbol{\Theta}^{\mathrm{T}}_{\Gamma',i'}(t)\,\mathrm{d}t + \int_0^{T_p}\boldsymbol{\Theta}^{\mathrm{T}}_{\Gamma^*,i'}(t)\,\mathrm{d}t + \int_0^{T_p}\boldsymbol{\Theta}^{\mathrm{T}}_{\Gamma_0,i'}(t)\,\mathrm{d}t \\
&= \boldsymbol{c}^{\mathrm{T}}_{\Gamma',i'} + \boldsymbol{c}^{\mathrm{T}}_{\Gamma^*,i'} + \boldsymbol{c}^{\mathrm{T}}_{\Gamma_0,i'},
\end{aligned} \tag{D.6}$$

where

$$\boldsymbol{\Theta}^{\mathrm{T}}_{\Gamma',i'}(t) = \boldsymbol{\Gamma}'^{\mathrm{T}}(t)\boldsymbol{Q}\boldsymbol{e}^{\boldsymbol{F}(t-t'_{p,i})}\boldsymbol{G}_p h(t - t'_{p,i}) \tag{D.7}$$

$$\boldsymbol{\Theta}^{\mathrm{T}}_{\Gamma^*,i'}(t) = \boldsymbol{\Gamma}^{*\mathrm{T}}(t)\boldsymbol{Q}\boldsymbol{e}^{\boldsymbol{F}(t-t'_{p,i})}\boldsymbol{G}_p h(t - t'_{p,i}) \tag{D.8}$$

$$\boldsymbol{\Theta}^{\mathrm{T}}_{\Gamma_0,i'}(t) = \boldsymbol{\Gamma}^{\mathrm{T}}_0(t)\boldsymbol{Q}\boldsymbol{e}^{\boldsymbol{F}(t-t'_{p,i})}\boldsymbol{G}_p h(t - t'_{p,i}). \tag{D.9}$$

First consider (D.7), which is further decomposed into each phase separately,

$$\begin{aligned}\boldsymbol{\Theta}^{\mathrm{T}}_{\Gamma',p_2,i'}(t) &= \sum_{j=1}^{n_{p_2}} \Delta u_{p_2,j} \boldsymbol{G}^{\mathrm{T}}_{p,2} (\boldsymbol{F}^{-1})^{\mathrm{T}} \left(\boldsymbol{e}^{\boldsymbol{F}(t-t'_{p_2,j})} - \mathbf{I}_{n_x}\right)^{\mathrm{T}} h(t-t'_{p_2,j}) \boldsymbol{Q} \boldsymbol{e}^{\boldsymbol{F}(t-t'_{p,i})} \boldsymbol{G}_p h(t-t'_{p,i}) \\ &= \sum_{j=1}^{n_{p_2}} \left(\Delta u_{p_2,j} \boldsymbol{G}^{\mathrm{T}}_{p_2} (\boldsymbol{F}^{-1})^{\mathrm{T}} \left(\boldsymbol{e}^{\boldsymbol{F}(t-t'_{p_2,j})} - \mathbf{I}_{n_x}\right)^{\mathrm{T}} \boldsymbol{Q} \boldsymbol{e}^{\boldsymbol{F}(t-t'_{p,i})} h(t-t'_{ij})\right) \boldsymbol{G}_p, \quad \text{(D.10)}\end{aligned}$$

where p_2 is a particular phase of (D.3) and $t'_{ij} = \max\{t'_{p,i}, t'_{p_2,j}\}$. Note that $\boldsymbol{\Theta}_{\Gamma',i'} = \boldsymbol{\Theta}_{\Gamma',a,i'} + \boldsymbol{\Theta}_{\Gamma',b,i'} + \boldsymbol{\Theta}_{\Gamma',c,i'}$. Its integral

$$\boldsymbol{c}^{\mathrm{T}}_{\Gamma',p_2,i'} = \sum_{j=1}^{n_{p_2}} \left(\boldsymbol{G}^{\mathrm{T}}_{p_2} (\boldsymbol{F}^{-1})^{\mathrm{T}} \Delta u_{p_2,j} \int_0^{T_p} \left(\boldsymbol{e}^{\boldsymbol{F}(t-t'_{p_2,j})} - \mathbf{I}_{n_x}\right)^{\mathrm{T}} \boldsymbol{Q} \boldsymbol{e}^{\boldsymbol{F}(t-t'_{p,i})} h(t-t'_{ij})\, \mathrm{d}t\right) \boldsymbol{G}_p$$

is split into two parts as

$$\begin{aligned}\boldsymbol{c}^{\mathrm{T}}_{\Gamma',p_2,i'} = \sum_{j=1}^{n_{p_2}} \Bigg(\boldsymbol{G}^{\mathrm{T}}_{p_2} (\boldsymbol{F}^{-1})^{\mathrm{T}} \Delta u_{p_2,j} \Bigg(&\int_{t'_{ij}}^{T_p} \boldsymbol{e}^{\boldsymbol{F}^{\mathrm{T}}(t-t'_{p_2,j})} \boldsymbol{Q} \boldsymbol{e}^{\boldsymbol{F}(t-t'_{p,i})}\, \mathrm{d}t \\ &- \int_{t'_{ij}}^{T_p} \boldsymbol{Q} \boldsymbol{e}^{\boldsymbol{F}(t-t'_{p,i})}\, \mathrm{d}t\Bigg)\Bigg) \boldsymbol{G}_p\end{aligned}$$

which, following a time shift forward with t'_{ij} and using algebraic manipulations similar to those used in the derivation of (5.43), becomes

$$\begin{aligned}\boldsymbol{c}^{\mathrm{T}}_{\Gamma',p_2,i'} = \sum_{j=1}^{n_{p_2}} \Bigg(\boldsymbol{G}^{\mathrm{T}}_{p_2} (\boldsymbol{F}^{-1})^{\mathrm{T}} \Delta u_{p_2,j} \Bigg(&\boldsymbol{e}^{\boldsymbol{F}^{\mathrm{T}}(t'_{ij}-t'_{p_2,j})} \boldsymbol{\Xi}(T_p - t'_{ij}) \\ &- \int_0^{T_p - t'_{ij}} \boldsymbol{Q} \boldsymbol{e}^{\boldsymbol{F}t}\, \mathrm{d}t\Bigg) \boldsymbol{e}^{\boldsymbol{F}(t'_{ij}-t'_{p,i})}\Bigg) \boldsymbol{G}_p,\end{aligned} \quad \text{(D.11)}$$

where $\boldsymbol{\Xi}$ is defined in (5.41). According to [58], the second integral can be calculated using

$$\boldsymbol{W}(\Delta T) = \int_0^{\Delta T} \boldsymbol{Q} \boldsymbol{e}^{\boldsymbol{F}t}\, \mathrm{d}t \quad \text{(D.12)}$$

where

$$\begin{bmatrix} \mathbf{I}_{n_x} & \boldsymbol{W}(\Delta T) \\ \mathbf{0}_{n_x \times n_x} & \boldsymbol{M}(\Delta T) \end{bmatrix} = \boldsymbol{e}^{\begin{bmatrix} \mathbf{0}_{n_x \times n_x} & \boldsymbol{Q} \\ \mathbf{0}_{n_x \times n_x} & \boldsymbol{F} \end{bmatrix} \Delta T} = \begin{bmatrix} \mathbf{I}_{n_x} & \int_0^{\Delta T} \boldsymbol{Q} \boldsymbol{e}^{\boldsymbol{F}t}\, \mathrm{d}t \\ \mathbf{0}_{n_x \times n_x} & \boldsymbol{e}^{\boldsymbol{F}\Delta T} \end{bmatrix}.$$

Using (D.12), (D.11) becomes

$$\boldsymbol{c}^{\mathrm{T}}_{\Gamma',p_2,i'} = \sum_{j=1}^{n_{p_2}} \left(\boldsymbol{G}^{\mathrm{T}}_{p_2} (\boldsymbol{F}^{-1})^{\mathrm{T}} \Delta u_{p_2,j} \left(\boldsymbol{e}^{\boldsymbol{F}^{\mathrm{T}}(t'_{ij}-t'_{p_2,j})} \boldsymbol{\Xi}(T_p - t'_{ij}) - \boldsymbol{W}(T_p - t'_{ij})\right) \boldsymbol{e}^{\boldsymbol{F}(t'_{ij}-t'_{p,i})}\right) \boldsymbol{G}_p. \quad \text{(D.13)}$$

Next consider (D.8), which has the same structure as (D.7) and can thus be stated as

$$\boldsymbol{c}^{\mathrm{T}}_{\Gamma^*,p_2,i'} = \sum_{j=1}^{n^*_{p_2}} \left(\boldsymbol{G}^{\mathrm{T}}_{p_2} (\boldsymbol{F}^{-1})^{\mathrm{T}} \Delta u^*_{p_2,j} \left(\boldsymbol{e}^{\boldsymbol{F}^{\mathrm{T}}(t'^*_{ij}-t^*_{p_2,j})} \boldsymbol{\Xi}(T_p - t'^*_{ij}) - \boldsymbol{W}(T_p - t'^*_{ij})\right) \boldsymbol{e}^{\boldsymbol{F}(t'^*_{ij}-t'_{p,i})}\right) \boldsymbol{G}_p, \quad \text{(D.14)}$$

where $t'^{*}_{ij} = \max\{t'_{p,i}, t^{*}_{p_2,j}\}$.

Lastly, consider (D.9), which can be re-written as

$$\boldsymbol{\Theta}^{\mathrm{T}}_{\Gamma_0,i'}(t) = \Delta \boldsymbol{u}^{\mathrm{T}}_{abc,0} \boldsymbol{G}^{\mathrm{T}} (\boldsymbol{F}^{-1})^{\mathrm{T}} \left(\boldsymbol{e}^{\boldsymbol{F}t} - \mathbf{I}_{n_x}\right)^{\mathrm{T}} \boldsymbol{Q} \boldsymbol{e}^{\boldsymbol{F}(t-t'_{p,i})} \boldsymbol{G}_p h(t - t'_{p,i}). \tag{D.15}$$

Its integral is also split into two parts as

$$\boldsymbol{c}^{\mathrm{T}}_{\Gamma_0,i'} = \Delta \boldsymbol{u}^{\mathrm{T}}_{abc,0} \boldsymbol{G}^{\mathrm{T}} (\boldsymbol{F}^{-1})^{\mathrm{T}} \left(\int_{t'_{p,i}}^{T_p} \boldsymbol{e}^{\boldsymbol{F}^{\mathrm{T}}t} \boldsymbol{Q} \boldsymbol{e}^{\boldsymbol{F}(t-t'_{p,i})} \, \mathrm{d}t - \int_{t'_{p,i}}^{T_p} \boldsymbol{Q} \boldsymbol{e}^{\boldsymbol{F}(t-t'_{p,i})} \, \mathrm{d}t \right) \boldsymbol{G}_p,$$

and following a time shift forward by $t'_{p,i}$, becomes

$$\boldsymbol{c}^{\mathrm{T}}_{\Gamma_0,i'} = \Delta \boldsymbol{u}^{\mathrm{T}}_{abc,0} \boldsymbol{G}^{\mathrm{T}} (\boldsymbol{F}^{-1})^{\mathrm{T}} \left(\boldsymbol{e}^{\boldsymbol{F}^{\mathrm{T}}t'_{p,i}} \int_{0}^{T_p - t'_{p,i}} \boldsymbol{e}^{\boldsymbol{F}^{\mathrm{T}}t} \boldsymbol{Q} \boldsymbol{e}^{\boldsymbol{F}t} \, \mathrm{d}t - \int_{0}^{T_p - t'_{p,i}} \boldsymbol{Q} \boldsymbol{e}^{\boldsymbol{F}t} \, \mathrm{d}t \right) \boldsymbol{G}_p. \tag{D.16}$$

Using (5.41) and (D.12), (D.16) becomes

$$\boldsymbol{c}^{\mathrm{T}}_{\Gamma_0,i'} = \Delta \boldsymbol{u}^{\mathrm{T}}_{abc,0} \boldsymbol{G}^{\mathrm{T}} (\boldsymbol{F}^{-1})^{\mathrm{T}} \left(\boldsymbol{e}^{\boldsymbol{F}^{\mathrm{T}}t'_{p,i}} \boldsymbol{\Xi}(T_p - t'_{p,i}) - \boldsymbol{W}(T_p - t'_{p,i}) \right) \boldsymbol{G}_p. \tag{D.17}$$

Appendix E

Definiteness of the Hessian

Consider the Hessian $\boldsymbol{H} = \boldsymbol{V} + \boldsymbol{R}$, where

$$\boldsymbol{V} = \int_0^{T_p} \boldsymbol{\Phi}^{\mathrm{T}}(t)\boldsymbol{Q}\boldsymbol{\Phi}(t)\,\mathrm{d}t,$$

and $\boldsymbol{R} \in \mathbb{R}^{n_{\mathrm{sw}} \times n_{\mathrm{sw}}}$ and $\boldsymbol{Q} \in \mathbb{R}^{n_x \times n_x}$ are positive (semi)definite matrices. Recall that $\boldsymbol{\Phi} \in \mathbb{R}^{n_x \times n_{\mathrm{sw}}}$ is defined as

$$\begin{aligned}\boldsymbol{\Phi}(t) = \Big[& \boldsymbol{e}^{\boldsymbol{F}(t-t'_{a,1})}\boldsymbol{G}_a h(t-t'_{a,1}) \;\cdots\; \boldsymbol{e}^{\boldsymbol{F}(t-t'_{a,n_a})}\boldsymbol{G}_a h(t-t'_{a,n_a}) \\ & \boldsymbol{e}^{\boldsymbol{F}(t-t'_{b,1})}\boldsymbol{G}_b h(t-t'_{b,1}) \;\cdots\; \boldsymbol{e}^{\boldsymbol{F}(t-t'_{b,n_b})}\boldsymbol{G}_b h(t-t'_{b,n_b}) \\ & \boldsymbol{e}^{\boldsymbol{F}(t-t'_{c,1})}\boldsymbol{G}_c h(t-t'_{c,1}) \;\cdots\; \boldsymbol{e}^{\boldsymbol{F}(t-t'_{c,n_c})}\boldsymbol{G}_c h(t-t'_{c,n_c}) \Big].\end{aligned}$$

Consider the case where the penality matrix $\boldsymbol{R}$ is zero, thus $\boldsymbol{H} = \boldsymbol{V}$. To show the Hessian is at least positive semidefinite, note that

$$\|\boldsymbol{\Phi}(t)\boldsymbol{\Lambda}\|_{\boldsymbol{Q}} \geq 0$$

for any $\boldsymbol{\Lambda} \in \mathbb{R}^{n_{\mathrm{sw}}}$ (and given that $\boldsymbol{Q}$ is positive semidefinite). Then, it follows from [76, Theorem 7.1.5] that

$$\int_0^{T_p} \|\boldsymbol{\Phi}(t)\boldsymbol{\Lambda}\|_{\boldsymbol{Q}}\mathrm{d}t \geq 0, \tag{E.1}$$

thus showing that the Hessian $\boldsymbol{H}$ is at least positive semidefinite.

Furthermore, if $\boldsymbol{Q}$ is positive definite, it can be shown there exists an interval on $[0, T_p]$ that guarantees

$$\int_0^{T_p} \|\boldsymbol{\Phi}(t)\boldsymbol{\Lambda}\|_{\boldsymbol{Q}}\mathrm{d}t > 0 \tag{E.2}$$

if no switching transitions $t'_{p,i}$ occur at the same time. The proof is omitted, but it involves noting that only one column of $\boldsymbol{\Phi}$ becomes nonzero at any given time. Thus, the Hessian $\boldsymbol{H}$ is guaranteed to be positive definite if no switching transitions $t'_{p,i}$ occur at the same time (and assuming $\boldsymbol{Q}$ is positive definite).

Note that having switching transitions $t'_{p,i}$ in the same phase occur at the same time leads to at least two (identical) columns of $\boldsymbol{\Phi}$ becoming nonzero at the same instant. This leads to the Hessian $\boldsymbol{H}$ being positive semidefinite, since it will have identical columns. It can be deduced that having switching transitions $t'_{p,i}$ in the same phase that are close together will result in ill-conditioning.

Finally, consider the case where the penality matrix $\boldsymbol{R}$ is positive definite (meaning all switching transition modifications are penalized). It then follows that the Hessian $\boldsymbol{H}$ is always positive definite since

$$\boldsymbol{\Lambda}^{\mathrm{T}}(\boldsymbol{V}+\boldsymbol{R})\boldsymbol{\Lambda} > 0 \tag{E.3}$$

even if $\boldsymbol{V}$ is positive semidefinite.

Appendix F

Manipulations of Small-Signal Neutral-Point Error

Recall the integrals

$$\boldsymbol{m}_1^{\mathrm{T}}(t)\boldsymbol{\Lambda} + \boldsymbol{\Lambda}^{\mathrm{T}}\boldsymbol{M}(t)\boldsymbol{\Lambda} = \frac{1}{2C_d}\int_0^t \tilde{\boldsymbol{x}}^{\mathrm{T}}(\tau,\lambda_{p,i})\boldsymbol{T}^{\mathrm{T}}\tilde{\boldsymbol{u}}'_{abc}(\tau,\lambda_{p,i})\,\mathrm{d}\tau \tag{F.1}$$

$$\theta(t) + \boldsymbol{m}_2^{\mathrm{T}}(t)\boldsymbol{\Lambda} = \frac{1}{2C_d}\int_0^t \tilde{\boldsymbol{x}}^{\mathrm{T}}(\tau,\lambda_{p,i})\boldsymbol{T}^{\mathrm{T}}\boldsymbol{u}'_{abc}(\tau)\,\mathrm{d}\tau \tag{F.2}$$

$$\boldsymbol{m}_3^{\mathrm{T}}(t)\boldsymbol{\Lambda} = \frac{1}{2C_d}\int_0^t \boldsymbol{x}^{*\mathrm{T}}(\tau)\boldsymbol{T}^{\mathrm{T}}\tilde{\boldsymbol{u}}'_{abc}(\tau,\lambda_{p,i})\,\mathrm{d}\tau \tag{F.3}$$

from Section 7.4.1. The transformation matrix $\boldsymbol{T} \in \mathbb{R}^{n_u \times n_x}$ calculates the three-phase converter current $\boldsymbol{i}_{abc}$ from the state vector $\boldsymbol{x}$. For convenience, the small-signal error of (7.36) is repeated here as

$$\tilde{\boldsymbol{x}}(t,\boldsymbol{\Lambda}) = \boldsymbol{e}^{\boldsymbol{F}t}\tilde{\boldsymbol{x}}_0 + \boldsymbol{\Phi}_n(t)\boldsymbol{\Lambda}, \tag{F.4}$$

where $\boldsymbol{\Phi}_n \in \mathbb{R}^{n_x \times n_{\mathrm{sw}}}$ is the input matrix,

$$\begin{aligned}\boldsymbol{\Phi}_n(t) = \Big[\; & \boldsymbol{e}^{\boldsymbol{F}(t-t'_{a,1})}\boldsymbol{G}_{a,1}h(t-t'_{a,1}) \;\cdots\; \boldsymbol{e}^{\boldsymbol{F}(t-t'_{a,n_a})}\boldsymbol{G}_{a,n_a}h(t-t'_{a,n_a}) \\ & \boldsymbol{e}^{\boldsymbol{F}(t-t'_{b,1})}\boldsymbol{G}_{b,1}h(t-t'_{b,1}) \;\cdots\; \boldsymbol{e}^{\boldsymbol{F}(t-t'_{b,n_b})}\boldsymbol{G}_{b,n_b}h(t-t'_{b,n_b}) \\ & \boldsymbol{e}^{\boldsymbol{F}(t-t'_{c,1})}\boldsymbol{G}_{c,1}h(t-t'_{c,1}) \;\cdots\; \boldsymbol{e}^{\boldsymbol{F}(t-t'_{c,nc})}\boldsymbol{G}_{c,n_c}h(t-t'_{c,n_c})\;\Big],\end{aligned} \tag{F.5}$$

and $\boldsymbol{\Lambda} \in \mathbb{R}^{n_{\mathrm{sw}}}$ is the strength vector,

$$\boldsymbol{\Lambda} = \begin{bmatrix}\lambda_{a,1} & \cdots & \lambda_{a,n_a} & \lambda_{b,1} & \cdots & \lambda_{b,n_b} & \lambda_{c,1} & \cdots & \lambda_{c,n_c}\end{bmatrix}^{\mathrm{T}}.$$

Note the term ϕ of (7.36) has been ignored in (F.4), since preliminary simulations show that it has a small influence on the integrals of (F.1)–(F.3) (and omitting it moderately reduces the computational burden). Finally, recall from Section 7.3.2 that

$$\boldsymbol{u}'_{abc}(t) = \begin{bmatrix}u'_a(t)\\ u'_b(t)\\ u'_c(t)\end{bmatrix} = \begin{bmatrix}u'_{a,0} + \sum_{i=1}^{n_a}\Delta u'_{a,i}h(t-t'_{a,i})\\ u'_{b,0} + \sum_{i=1}^{n_b}\Delta u'_{b,i}h(t-t'_{b,i})\\ u'_{c,0} + \sum_{i=1}^{n_c}\Delta u'_{c,i}h(t-t'_{c,i})\end{bmatrix} \tag{F.6}$$

and

$$\tilde{\boldsymbol{u}}'_{abc}(t,\lambda_{p,i}) = \begin{bmatrix}\tilde{u}'_a(t,\lambda_{a,i})\\ \tilde{u}'_b(t,\lambda_{b,i})\\ \tilde{u}'_c(t,\lambda_{c,i})\end{bmatrix} = \begin{bmatrix}\sum_{i=1}^{n_a}\frac{\Delta u'_{a,i}}{\Delta u_{a,i}}\lambda_{a,i}\delta(t-t'_{a,i})\\ \sum_{i=1}^{n_b}\frac{\Delta u'_{b,i}}{\Delta u_{b,i}}\lambda_{b,i}\delta(t-t'_{b,i})\\ \sum_{i=1}^{n_c}\frac{\Delta u'_{c,i}}{\Delta u_{c,i}}\lambda_{c,i}\delta(t-t'_{c,i})\end{bmatrix}. \tag{F.7}$$

First, consider the integral of (F.1). By using (F.7), the term $\boldsymbol{T}^{\mathrm{T}}\tilde{\boldsymbol{u}}'_{abc}$ can be written in terms of $\boldsymbol{\Lambda}$ as

$$\boldsymbol{T}^{\mathrm{T}}\tilde{\boldsymbol{u}}'_{abc}(\tau,\lambda_{p,i}) = \boldsymbol{\Psi}(t)\boldsymbol{\Lambda} \tag{F.8}$$

where $\boldsymbol{\Psi} \in \mathbb{R}^{n_x \times n_{\mathrm{sw}}}$ is defined as

$$\begin{aligned}\boldsymbol{\Psi}(t) = \Big[&\frac{\Delta u'_{a,1}}{\Delta u_{a,1}}\boldsymbol{T}_a\delta(t-t'_{a,1}) \ \cdots \ \frac{\Delta u'_{a,n_a}}{\Delta u_{a,n_a}}\boldsymbol{T}_a\delta(t-t'_{a,n_a}) \\ &\frac{\Delta u'_{b,1}}{\Delta u_{b,1}}\boldsymbol{T}_b\delta(t-t'_{b,1}) \ \cdots \ \frac{\Delta u'_{b,n_b}}{\Delta u_{b,n_b}}\boldsymbol{T}_b\delta(t-t'_{b,n_b}) \\ &\frac{\Delta u'_{c,1}}{\Delta u_{c,1}}\boldsymbol{T}_c\delta(t-t'_{c,1}) \ \cdots \ \frac{\Delta u'_{c,n_c}}{\Delta u_{c,n_c}}\boldsymbol{T}_c\delta(t-t'_{c,n_c}) \Big],\end{aligned} \tag{F.9}$$

and where $\boldsymbol{T}_a = \boldsymbol{T}^{\mathrm{T}}[1\ 0\ 0]^{\mathrm{T}}$, $\boldsymbol{T}_b = \boldsymbol{T}^{\mathrm{T}}[0\ 1\ 0]^{\mathrm{T}}$, and $\boldsymbol{T}_c = \boldsymbol{T}^{\mathrm{T}}[0\ 0\ 1]^{\mathrm{T}}$ have been introduced. Inserting (F.4) and (F.8) into the integral of (F.1) yields

$$\boldsymbol{m}_1^{\mathrm{T}}(t)\boldsymbol{\Lambda} + \boldsymbol{\Lambda}^{\mathrm{T}}\boldsymbol{M}(t)\boldsymbol{\Lambda}, \tag{F.10}$$

where $\boldsymbol{m}_1 \in \mathbb{R}^{n_{\mathrm{sw}}}$ and $\boldsymbol{M} \in \mathbb{R}^{n_{\mathrm{sw}} \times n_{\mathrm{sw}}}$ are defined as

$$\boldsymbol{m}_1^{\mathrm{T}}(t) = \tilde{\boldsymbol{x}}_0^{\mathrm{T}}\frac{1}{2C_d}\int_0^t \boldsymbol{e}^{\boldsymbol{F}^{\mathrm{T}}\tau}\boldsymbol{\Psi}(\tau)\,\mathrm{d}\tau \tag{F.11}$$

$$\boldsymbol{M}(t) = \frac{1}{2C_d}\int_0^t \boldsymbol{\Phi}_n^{\mathrm{T}}(\tau)\boldsymbol{\Psi}(\tau)\,\mathrm{d}\tau. \tag{F.12}$$

Consider the i'th entry of $\boldsymbol{m}_{1,i'}$, which [after inserting (F.9)] is given as

$$\begin{aligned}\boldsymbol{m}_{1,i'}^{\mathrm{T}}(t) &= \tilde{\boldsymbol{x}}_0^{\mathrm{T}}\frac{1}{2C_d}\frac{\Delta u'_{p,i}}{\Delta u_{p,i}}\int_0^t \boldsymbol{e}^{\boldsymbol{F}^{\mathrm{T}}\tau}\boldsymbol{T}_p\delta(\tau-t'_{p,i})\,\mathrm{d}\tau \\ &= \frac{1}{2C_d}\frac{\Delta u'_{p,i}}{\Delta u_{p,i}}\tilde{\boldsymbol{x}}_0^{\mathrm{T}}\boldsymbol{e}^{\boldsymbol{F}^{\mathrm{T}}t'_{p,i}}\boldsymbol{T}_p h(\tau-t'_{p,i}).\end{aligned} \tag{F.13}$$

Similarly, consider the (i',j')th entry of $\boldsymbol{M}$, which [after inserting (F.5) and (F.9)] follows as

$$\boldsymbol{M}_{(i',j')}(t) = \frac{1}{2C_d}\boldsymbol{G}_{p_1,i}^{\mathrm{T}}\int_0^t \boldsymbol{e}^{\boldsymbol{F}^{\mathrm{T}}(\tau-t'_{p_1,i})}h(\tau-t'_{p_1,i})\delta(\tau-t'_{p_2,j})\,\mathrm{d}\tau\,\boldsymbol{T}_{p_2}\frac{\Delta u'_{p_2,j}}{\Delta u_{p_2,j}} \tag{F.14}$$

$$= \begin{cases} \boldsymbol{G}_{p_1,i}^{\mathrm{T}}\boldsymbol{e}^{\boldsymbol{F}^{\mathrm{T}}(t'_{p_2,j}-t'_{p_1,i})}h(t'_{p_2,j}-t'_{p_1,i})h(t-t'_{p_2,j})\boldsymbol{T}_{p_2}\frac{\Delta u'_{p_2,j}}{\Delta u_{p_2,j}} & \text{if } t'_{p_2,j} \neq t'_{p_1,i} \\ \frac{1}{2}\boldsymbol{G}_{p_1,i}^{\mathrm{T}}\boldsymbol{e}^{\boldsymbol{F}^{\mathrm{T}}(t'_{p_2,j}-t'_{p_1,i})}h(t-t'_{p_2,j})\boldsymbol{T}_{p_2}\frac{\Delta u'_{p_2,j}}{\Delta u_{p_2,j}} & \text{else.}\end{cases} \tag{F.15}$$

Note that the sifting property of the impulse function assumes a function to be continuous where the impulse occurs, which is not the case when $t'_{p_2,j} = t'_{p_1,i}$ in (F.14). For this case, it is assumed that half of the value is sifted.

Next, consider the integral of (F.2), which can be written as

$$\theta(t) + \boldsymbol{m}_2^{\mathrm{T}}(t)\boldsymbol{\Lambda} \tag{F.16}$$

where $\theta \in \mathbb{R}$ and $\boldsymbol{m}_2 \in \mathbb{R}^{n_{\mathrm{sw}}}$ are defined as

$$\theta(t) = \tilde{\boldsymbol{x}}_0^{\mathrm{T}}\frac{1}{2C_d}\int_0^t \boldsymbol{e}^{\boldsymbol{F}^{\mathrm{T}}\tau}\left(\boldsymbol{T}_a u'_a(\tau) + \boldsymbol{T}_b u'_b(\tau) + \boldsymbol{T}_c u'_c(\tau)\right)\,\mathrm{d}\tau \tag{F.17}$$

$$\boldsymbol{m}_2^{\mathrm{T}} = \frac{1}{2C_d}\int_0^t \left(\boldsymbol{T}_a^{\mathrm{T}}u'_a(\tau) + \boldsymbol{T}_b^{\mathrm{T}}u'_b(\tau) + \boldsymbol{T}_c^{\mathrm{T}}u'_c(\tau)\right)\boldsymbol{\Phi}_n(\tau)\,\mathrm{d}\tau. \tag{F.18}$$

Consider θ, which is decomposed into three phases as $\theta = \theta_a + \theta_b + \theta_c$, where phase p is

$$\begin{aligned}\theta_p(t) &= \tilde{\boldsymbol{x}}_0^{\mathrm{T}} \frac{1}{2C_d} \int_0^t \boldsymbol{e}^{\boldsymbol{F}^{\mathrm{T}}\tau} \boldsymbol{T}_p \left(u'_{p,0} + \sum_{i=1}^{n_p} \Delta u'_{p,i} h(\tau - t'_{p,i}) \right) \mathrm{d}\tau \\ &= \tilde{\boldsymbol{x}}_0^{\mathrm{T}} \left(\boldsymbol{F}^{-1}\right)^{\mathrm{T}} \left((\boldsymbol{e}^{\boldsymbol{F}^{\mathrm{T}}t} - \mathbf{I}_{n_x}) u'_{p,0} + \sum_{i=1}^{n_p} (\boldsymbol{e}^{\boldsymbol{F}^{\mathrm{T}}t} - \boldsymbol{e}^{\boldsymbol{F}^{\mathrm{T}}t'_{p,i}}) \Delta u'_{p,i} h(t - t'_{p,i}) \right) \boldsymbol{T}_p. \end{aligned} \tag{F.19}$$

Similarly, $\boldsymbol{m}_2$ is decomposed into three phases as $\boldsymbol{m}_2 = \boldsymbol{m}_{2,a} + \boldsymbol{m}_{2,b} + \boldsymbol{m}_{2,c}$, where a particular phase p_2 is

$$\boldsymbol{m}_{2,p_2}^{\mathrm{T}} = \frac{1}{2C_d} \int_0^t \boldsymbol{T}_{p_2}^{\mathrm{T}} \left(u'_{p_2,0} + \sum_{j=1}^{n_{p_2}} \Delta u'_{p_2,j} h(\tau - t'_{p_2,j}) \right) \boldsymbol{\Phi}_n(\tau) \, \mathrm{d}\tau. \tag{F.20}$$

The i'th entry of $\boldsymbol{m}_{2,p_2}$ is

$$\begin{aligned}\boldsymbol{m}_{2,p_2,i'}^{\mathrm{T}} &= \frac{1}{2C_d} \int_0^t \boldsymbol{T}_p^{\mathrm{T}} \left(u'_{p_2,0} + \sum_{j=1}^{n_{p_2}} \Delta u'_{p_2,j} h(\tau - t'_{p_2,j}) \right) \boldsymbol{e}^{\boldsymbol{F}(\tau - t'_{p,i})} h(\tau - t'_{p,i}) \, \mathrm{d}\tau \, \boldsymbol{G}_{p,i} \\ &= \frac{1}{2C_d} \boldsymbol{T}_p^{\mathrm{T}} \Bigg((\boldsymbol{e}^{\boldsymbol{F}(t - t'_{p,i})} - \mathbf{I}_{n_x}) u'_{p_2,0} h(\tau - t'_{p,i}) \\ &\quad + \sum_{j=1}^{n_{p_2}} (\boldsymbol{e}^{\boldsymbol{F}(t - t'_{p,i})} - \boldsymbol{e}^{\boldsymbol{F}(t'_{p,ij} - t'_{p,i})}) \Delta u'_{p_2,j} h(t - t'_{p,ij}) \boldsymbol{F}^{-1} \boldsymbol{G}_{p,i} \Bigg) \end{aligned} \tag{F.21}$$

where $t'_{p,ij} = \max\{t'_{p,i}, t'_{p_2,j}\}$.

Finally, consider the integral of (F.3), which with (F.8) can be written as

$$\boldsymbol{m}_3^{\mathrm{T}}(t) \boldsymbol{\Lambda} \tag{F.22}$$

where $\boldsymbol{m}_3 \in \mathbb{R}^{n_{\mathrm{sw}}}$ is

$$\boldsymbol{m}_3^{\mathrm{T}}(t) = \frac{1}{2C_d} \int_0^t \boldsymbol{x}^{*\mathrm{T}}(\tau) \boldsymbol{\Psi}(\tau) \, \mathrm{d}\tau. \tag{F.23}$$

The i'th entry of $\boldsymbol{m}_3$ [after inserting (F.9)] follows as

$$\begin{aligned}\boldsymbol{m}_{3,i'}^{\mathrm{T}}(t) &= \frac{1}{2C_d} \frac{\Delta u'_{p,i}}{\Delta u_{p,i}} \int_0^t \boldsymbol{x}^{*\mathrm{T}}(\tau) \delta(\tau - t'_{p,i}) \, \mathrm{d}\tau \, \boldsymbol{T}_p \\ &= \frac{1}{2C_d} \frac{\Delta u'_{p,i}}{\Delta u_{p,i}} \boldsymbol{x}^{*\mathrm{T}}(t'_{p,i}) h(t - t'_{p,i}) \, \boldsymbol{T}_p. \end{aligned} \tag{F.24}$$

Bibliography

[1] T. Noguchi, H. Tomiki, S. Kondo, and I. Takahashi, "Direct power control of PWM converter without power-source voltage sensors," *IEEE Transactions on Industry Applications*, vol. 34, no. 3, pp. 473–479, Jun. 1998.

[2] M. Malinowski, M. Kazmierkowski, and A. Trzynadlowski, "A comparative study of control techniques for PWM rectifiers in AC adjustable speed drives," *IEEE Transactions on Power Electronics*, vol. 18, no. 6, pp. 1390–1396, Nov. 2003.

[3] T. Geyer, N. Oikonomou, G. Papafotiou, and F. D. Kieferndorf, "Model predictive pulse pattern control," *IEEE Transactions on Industry Applications*, vol. 48, no. 2, pp. 663–676, Mar. 2012.

[4] T. Geyer, *Model predictive control of high power converters and industrial drives*. Chichester, UK: John Wiley & Sons, Ltd, Nov. 2016.

[5] M. D. Dorfling, H. d. T. Mouton, and T. Geyer, "Model predictive pulse pattern control based on small-signal pulse pattern optimization," Patent application EP 19 202 732.4 - 1201.

[6] D. P. Bertsekas, *Nonlinear programming*, 3rd ed. Belmont, Massachusetts: Athena Scientific, 2016.

[7] J. Nocedal and S. J. Wright, *Numerical optimization*, 2nd ed., ser. Springer Series in Operations Research and Financial Engineering. New York: Springer, 2006.

[8] M. S. Bazaraa, H. D. Sherali, and C. M. Shetty, *Nonlinear programming: theory and algorithms*, 3rd ed. Hoboken, N.J: Wiley-Interscience, 2006.

[9] S. P. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge, UK ; New York: Cambridge University Press, 2004.

[10] Y. Nesterov, *Lectures on convex optimization*, ser. Springer Optimization and Its Applications. Cham: Springer International Publishing, 2018, vol. 137.

[11] A. Beck, *Introduction to nonlinear optimization: theory, algorithms, and applications with Matlab*. Philadelphia, PA: Society for Industrial and Applied Mathematics, Oct. 2014.

[12] P. M. Pardalos, "Global optimization algorithms for linearly constrained indefinite quadratic problems," *Computers & Mathematics with Applications*, vol. 21, no. 6-7, pp. 87–97, 1991.

[13] Y. Nesterov and A. Nemirovskii, *Interior-point polynomial algorithms in convex programming*. Society for Industrial and Applied Mathematics, Jan. 1994.

[14] S. Richter, "Computational complexity certification of gradient methods for real-time model predictive control," PhD thesis, ETH Zurich, 2012.

[15] O. Devolder, F. Glineur, and Y. Nesterov, "First-order methods of smooth convex optimization with inexact oracle," *Mathematical Programming*, vol. 146, no. 1-2, pp. 37–75, Aug. 2014.

[16] Y. Nesterov, "A method of solving a convex programming problem with convergence rate $O(\frac{1}{k^2})$," *Soviet Math. Dokl.*, 1983.

[17] G. Strang, *Introduction to linear algebra*, 5th ed. Wellesley, MA: Wellesley-Cambridge Press, 2016.

[18] A. Beck, *First-order methods in optimization*. Philadelphia, PA: Society for Industrial and Applied Mathematics, Oct. 2017.

[19] W. C. Duesterhoeft, M. W. Schulz, and E. Clarke, "Determination of instantaneous currents and voltages by means of alpha, beta, and zero components," *Transactions of the American Institute of Electrical Engineers*, vol. 70, no. 2, pp. 1248–1255, Jul. 1951.

[20] A. Nabae, I. Takahashi, and H. Akagi, "A new neutral-point-clamped PWM inverter," *IEEE Transactions on Industry Applications*, vol. IA-17, no. 5, pp. 518–523, Sep. 1981.

[21] Y. Huang, X. Yuan, J. Hu, and P. Zhou, "Modeling of VSC connected to weak grid for stability analysis of DC-link voltage control," *IEEE Journal of Emerging and Selected Topics in Power Electronics*, vol. 3, no. 4, pp. 1193–1204, Dec. 2015.

[22] *IEEE Std 519-1992: IEEE recommended practices and requirements for harmonic control in electrical power systems*, Apr. 1993.

[23] *IEEE Std 519-2014: IEEE recommended practices and requirements for harmonic control in electrical power systems*, Jun. 2014.

[24] G. S. Buja and G. B. Indri, "Optimal pulsewidth modulation for feeding AC motors," *IEEE Transactions on Industry Applications*, vol. IA-13, no. 1, pp. 38–44, Jan. 1977.

[25] A. K. Rathore, J. Holtz, and T. Boller, "Generalized optimal pulsewidth modulation of multilevel inverters for low-switching-frequency control of medium-voltage high-power industrial AC drives," *IEEE Transactions on Industrial Electronics*, vol. 60, no. 10, pp. 4215–4224, Oct. 2013.

[26] J. W. Nilsson and S. A. Riedel, *Electric circuits*, 10th ed. Boston: Pearson, 2015.

[27] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical Programming*, vol. 106, no. 1, pp. 25–57, Mar. 2006.

[28] A. Birth, T. Geyer, H. d. T. Mouton, and M. Dorfling, "Generalized three-level optimal pulse patterns with lower harmonic distortion," *IEEE Transactions on Power Electronics*, vol. 35, no. 6, pp. 5741–5752, Jun. 2020.

[29] D. G. Holmes and T. A. Lipo, *Pulse width modulation for power converters: principles and practice*. Hoboken, NJ: John Wiley, 2003.

[30] J. Shen, S. Schroder, H. Stagge, and R. W. De Doncker, "Impact of modulation schemes on the power capability of high-power converters with low pulse ratios," *IEEE Transactions on Power Electronics*, vol. 29, no. 11, pp. 5696–5705, Nov. 2014.

[31] R. Kalman, "Contributions to the theory of optimal control," *Boletin de la Sociedad Matematica Mexicana*, vol. 5, pp. 102–119, 1960.

[32] C. E. García, D. M. Prett, and M. Morari, "Model predictive control: theory and practice—a survey," *Automatica*, vol. 25, no. 3, pp. 335–348, May 1989.

[33] S. Qin and T. A. Badgwell, "A survey of industrial model predictive control technology," *Control Engineering Practice*, vol. 11, no. 7, pp. 733–764, Jul. 2003.

[34] J. Holtz and S. Stadtfeld, "A predictive controller for the stator current vector of AC machines fed from a switched voltage source," in *International Power Electronics Conference*, Tokyo, Japan, Apr. 1983, pp. 1665–1675.

[35] P. Karamanakos, E. Liegmann, T. Geyer, and R. Kennel, "Model predictive control of power electronic systems: methods, results, and challenges," *IEEE Open Journal of Industry Applications*, vol. 1, pp. 95–114, 2020.

[36] J. B. Rawlings, D. Q. Mayne, and M. M. Diehl, *Model predictive control: theory, computation, and design*, 2nd ed. Madison, Wisconsin: Nob Hill Publishing, 2017.

[37] L. Grüne and J. Pannek, *Nonlinear model predictive control: theory and algorithms*, 2nd ed., ser. Communications and Control Engineering. Cham: Springer International Publishing : Imprint: Springer, 2017.

[38] J. Rodríguez, J. Pontt, C. Silva, M. Salgado, S. Rees, U. Ammann, P. Lezana, R. Huerta, and P. Cortés, "Predictive control of three-phase inverter," *IET Electronics Letters*, vol. 40, no. 9, p. 561, 2004.

[39] E. Eiben, R. Ganian, D. Knop, S. Ordyniak, M. Pilipczuk, and M. Wrochna, "Integer programming and incidence treedepth," in *Integer Programming and Combinatorial Optimization*, A. Lodi and V. Nagarajan, Eds. Cham: Springer International Publishing, 2019, vol. 11480, pp. 194–204.

[40] T. Geyer and D. E. Quevedo, "Performance of multistep finite control set model predictive control for power electronics," *IEEE Transactions on Power Electronics*, vol. 30, no. 3, pp. 1633–1644, Mar. 2015.

[41] ——, "Multistep finite control set model predictive control for power electronics," *IEEE Transactions on Power Electronics*, vol. 29, no. 12, pp. 6836–6846, Dec. 2014.

[42] T. Dorfling, H. du Toit Mouton, T. Geyer, and P. Karamanakos, "Long-horizon finite-control-set model predictive control with nonrecursive sphere decoding on an FPGA," *IEEE Transactions on Power Electronics*, vol. 35, no. 7, pp. 7520–7531, Jul. 2020.

[43] J. Holtz and B. Beyer, "Off-line optimized synchronous pulsewidth modulation with on-line control during transients," *EPE Journal*, vol. 1, no. 3, pp. 193–200, Jan. 1991.

[44] J. Holtz and N. Oikonomou, "Synchronous optimal pulsewidth modulation and stator flux trajectory control for medium-voltage drives," *IEEE Transactions on Industry Applications*, vol. 43, no. 2, pp. 600–608, 2007.

[45] ——, "Estimation of the fundamental current in low-switching-frequency high dynamic medium-voltage drives," *IEEE Transactions on Industry Applications*, vol. 44, no. 5, pp. 1597–1605, Sep. 2008.

[46] T. Geyer, F. Kieferndorf, G. Papafotiou, and N. Oikonomou, "Method for controlling a converter," Patent EP 2 469 692.

[47] "Breakthrough drive brings high performance," *Steel Times International*, vol. 43, no. 3, p. 22, 2019.

[48] I. Pejcic, "Spectral predictive control in power electronics," Master's thesis, EPFL.

[49] P. Al Hokayem and I. Pejcic, "Model predictive damping of oscillations in an electrical converter system," Patent application EP 3 262 741.

[50] P. Hokayem, T. Geyer, and N. Oikonomou, "Active damping for model predictive pulse pattern control," in *2014 IEEE Energy Conversion Congress and Exposition (ECCE)*. Pittsburgh, PA, USA: IEEE, Sep. 2014, pp. 1220–1227.

[51] I. Pejcic, S. Almer, and H. Peyrl, "Voltage source converter MPC with optimized pulse patterns and minimization of integrated squared tracking error," in *2017 American Control Conference (ACC)*. Seattle, WA, USA: IEEE, May 2017, pp. 4069–4074.

[52] S. Almer, "Predictive pulse pattern control of an inverter with LCL filter: A nonlinear transformation approach," in *2017 IEEE 3rd International Future Energy Electronics Conference and ECCE Asia (IFEEC 2017 - ECCE Asia)*. Kaohsiung, Taiwan: IEEE, Jun. 2017, pp. 1031–1036.

[53] A. Birth, "Model predictive control of a medium-voltage grid-connected converter with LC filter using optimal pulse patterns with relaxed symmetry," Master's thesis, Stellenbosch University, 2020.

[54] G. C. Goodwin, S. F. Graebe, and M. E. Salgado, *Control system design*. Upper Saddle River, NJ: Prentice Hall, 2001.

[55] D. A. Neamen, *Microelectronics: circuit analysis and design*, 4th ed. New York: McGraw-Hill, 2010.

[56] H. d. T. Mouton, S. M. Cox, B. McGrath, L. Risbo, and B. Putzeys, "Small-signal analysis of naturally-sampled single-edge PWM control loops," *IEEE Transactions on Power Electronics*, vol. 33, no. 1, pp. 51–64, Jan. 2018.

[57] T. Mouton and T. Geyer, "Trajectory-based LQR control of a grid-connected converter with an LCL filter," *Proc. of the IFAC Conference on Nonlinear Model Predictive Control*, vol. 51, no. 20, pp. 273–278, 2018.

[58] C. Van Loan, "Computing integrals involving the matrix exponential," *IEEE Transactions on Automatic Control*, vol. 23, no. 3, pp. 395–404, Jun. 1978.

[59] T. Geyer and N. Oikonomou, "Model predictive pulse pattern control with very fast transient responses," in *2014 IEEE Energy Conversion Congress and Exposition (ECCE)*. Pittsburgh, PA, USA: IEEE, Sep. 2014, pp. 5518–5524.

[60] H. d. T. Mouton, "Natural balancing of three-level neutral-point-clamped PWM inverters," *IEEE Transactions on Industrial Electronics*, vol. 49, no. 5, pp. 1017–1025, Oct. 2002.

[61] J. Steinke, "Switching frequency optimal PWM control of a three-level inverter," *IEEE Transactions on Power Electronics*, vol. 7, no. 3, pp. 487–496, Jul. 1992.

[62] S. Ogasawara and H. Akagi, "Analysis of variation of neutral point potential in neutral-point-clamped voltage source PWM inverters," in *Conference Record of the 1993 IEEE Industry Applications Conference Twenty-Eighth IAS Annual Meeting*. Toronto, Ont., Canada: IEEE, 1993, pp. 965–970.

[63] C. Newton and M. Sumner, "Neutral point control for multi-level inverters: theory, design and operational limitations," in *IAS '97. Conference Record of the 1997 IEEE Industry Applications Conference Thirty-Second IAS Annual Meeting*, vol. 2. New Orleans, LA, USA: IEEE, 1997, pp. 1336–1343.

[64] J. Pou, R. Pindado, D. Boroyevich, and P. Rodriguez, "Evaluation of the low-frequency neutral-point voltage oscillations in the three-level inverter," *IEEE Transactions on Industrial Electronics*, vol. 52, no. 6, pp. 1582–1588, Dec. 2005.

[65] J. Holtz and N. Oikonomou, "Neutral point potential balancing algorithm at low modulation index for three-level inverter medium-voltage drives," *IEEE Transactions on Industry Applications*, vol. 43, no. 3, pp. 761–768, 2007.

[66] T. Geyer and V. Spudic, "Model predictive pulse pattern control with integrated balancing of the neutral point potential," in *2019 21st European Conference on Power Electronics and Applications (EPE '19 ECCE Europe)*. Genova, Italy: IEEE, Sep. 2019, pp. P.1–P.10.

[67] G. H. Golub and C. F. Van Loan, *Matrix computations*, 4th ed., ser. Johns Hopkins studies in the mathematical sciences. Baltimore: The Johns Hopkins University Press, 2013.

[68] M. Flynn, "On division by functional iteration," *IEEE Transactions on Computers*, vol. C-19, no. 8, pp. 702–706, Aug. 1970.

[69] R. L. Burden, J. D. Faires, and A. M. Burden, *Numerical analysis*, 10th ed. Boston, MA: Cengage Learning, 2016.

[70] S. Richter, T. Geyer, and M. Morari, "Resource-efficient gradient methods for model predictive pulse pattern control on an FPGA," *IEEE Transactions on Control Systems Technology*, vol. 25, no. 3, pp. 828–841, May 2017.

[71] A. Németh and S. Németh, "How to project onto an isotone projection cone," *Linear Algebra and its Applications*, vol. 433, no. 1, pp. 41–51, Jul. 2010.

[72] R. Jasinski, *Effective coding with VHDL: principles and best practice*. Cambridge, Massachusetts: The MIT Press, 2016.

[73] R. Cagienard, P. Grieder, E. Kerrigan, and M. Morari, "Move blocking strategies in receding horizon control," in *2004 43rd IEEE Conference on Decision and Control (CDC)*. Nassau, Bahamas: IEEE, 2004, pp. 2023–2028 Vol.2.

[74] C. Moler and C. Van Loan, "Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later," *SIAM Review*, vol. 45, no. 1, pp. 3–49, Jan. 2003.

[75] D. G. Zill and W. S. Wright, *Advanced engineering mathematics*. Burlington, MA: Jones & Bartlett Learning, 2014.

[76] R. G. Bartle and D. R. Sherbert, *Introduction to real analysis*, 4th ed. Hoboken, NJ: Wiley, 2011.